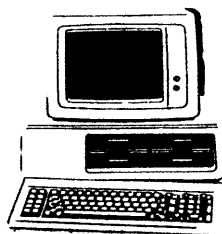


NOŠITEL
VYZNAMENÁNÍ
ZA BRANNOU
VÝCHOVU
II. STUPNĚ



mikroelektronika MIKROPROCESOROVÁ A VÝPOČETNÍ TECHNIKA HARDWARE & SOFTWARE

OBSAH

| | |
|--|----|
| Informovanost | 1 |
| Adaptér pro příjem teletextu pomocí mikropočítače | 2 |
| Teletext | 9 |
| Program teletext | 16 |
| Dlouhý spánek Gajany | 20 |
| CP/M, RAMDISK a řadič pružného disku | 22 |
| Televizní displej DIS-84 | 35 |
| Pripojenie tlačiarne s interfejsom IRPR/Centronics na Atari serial bus | 42 |
| Měřicí přípravek k A/C převodníku Digi 2 | 47 |
| RIDS | 53 |
| Skicák | 61 |
| ANALOG | 67 |
| ZX-Multitasking | 72 |
| DIATEM | 78 |

Amatérské radio Konstrukční příloha

Vydává ÚV Svazarmu, Opletalova 29,
116 31 Praha 1, tel. 22 25 49 ve Vydavatel-
ství NAŠE VOJSKO, Vladislavova 26,
113 66 Praha 1, tel. 26 06 51—7. Šéfredak-
tor ing. Jan Klábal, OK1UKA, zástupce
Luboš Kalousek, OK1FAC. Redakční rada:
předseda ing. J. T. Hyan, členové: RNDr.
V. Brunnhofer, OK1HAQ, V. Brzák,
OK1DDK, K. Donát, OK1DY, ing. O. Filippi,
A. Glanc, OK1GW, ing. F. Hanáček, P.
Horák, Z. Hradský, J. Hudec, OK1RE, ing.
J. Jaroš, ing. I. Kolmer, ing. F. Králík, J.
Kroupa, RNDr. L. Kryška, V. Němec, ing.
O. Petráček, OK1NB, ing. Z. Prošek, ing. F.
Smolík, OK1ASF, ing. F. Šimek, OK1FSI,
ing. M. Šredl, OK1NL, doc. ing. J. Vackář,
laureát st. ceny KG, J. Vorlíček.
Redakce: Jungmannova 24, 113 66 Praha
1, tel. 26 06 51—7, ing. Klábal, OK1UKA,
I. 354, L. Kalousek, OK1FAC, ing. P. Engel,
ing. J. Kellner, I. 353, ing. A. Myslík,
OK1AMY, P. Havliš, OK1PFM, I. 348,
sekretariát T. Trnková I. 355.
Rozšiřuje PNS, objednávky do zahraničí
vyřizuje rovněž PNS — ústřední expedice
a dovoz tisku, závod 01, administrace
vývozu tisku, Kačkova 9, 160 00 Praha 6.
V jednotkách ozbrojených sil rozšiřuje
Vydavatelství NAŠE VOJSKO, administra-
ce, Vladislavova 26, 113 66 Praha 1, tel.
26 06 51—7, I. 294.
Za původnost a správnost příspěvku ručí
autor. Rukopisy odevzdány tiskárně v říjnu
1988, tato Konstrukční příloha má vyjít po-
dle plánu v březnu 1989. Cena jednoho
výtisku 10 Kčs.
© Vydavatelství NAŠE VOJSKO, Praha.

INFORMOVANOST

Nedávno jsem si někde přečetl, že Japonci dali do prodeje komputer, který odpovídá na otázky dítěte — nechme stranou technický problém, jak se přístroji daří zanalyzovat dětskou řeč, pochopit smysl otázky, nalézt v paměti správnou odpověď a jak ji opět lidskou řečí přednést — ale co na to dítě? Ti co je mají, to znají z vlastní zkušenosti. „Dej mi pokoj“ nebo „co se pořád vyptáváš“ případně se stroze odpoví a ne vždy zcela správně. Kontakt dítěte s rodičem je tím sice zachován, ale ne zrovna tím nejlepším způsobem. Dítě bere rodiče jako neomylnou autoritu. Když se později dozví, že rodič neměl pravdu, nebo že nezná, je tato autorita poněkud otřesena, cit a láska zde však většinou zůstanou. Ale co komputer a vztah dítěte k jeho přesné a problém otázky zcela vyčerpávající odpovědi. Dítě se dovídá mnohem více, je rychleji vzdělané, ale je ochuzeno o citovou stránku vztahu k rodiči. Kromě toho vzniká vztah člověka ke stroji, k zařízení které člověku rozšiřuje obzor poznání, učí ho navíc vyjadřovací přesnosti, ale také tvořivému myšlení, jak otázku formulovat či jakou otázku položit, aby stroj překonal. Z člověka tak může vyrůstati vysoce inteligentní tvůrce dalších hodnot, člověk, který se třeba více uplatní v moderní technické společnosti, protože byl od mládí přesně a bohatě informován.

„Potřebujeme kvalitativně novou úroveň veřejné informovanosti i ve vojenské oblasti“ píše v listu IZVESTIJA Stanislav Kondrašov a pokračuje: „Bez ní nemůžeme počítat se stabilní důvěrou okolního světa vůči sovětské politice. Bez ní není možná seriózní veřejná diskuse v zemi o účelnosti a prioritách státního rozpočtu, o tom, jak velkou jeho část si můžeme dovolit věnovat na budování a udržování ozbrojených sil. Sovětský svaz se v roce 1987 na půdě OSN zavázal, že oznámí reálné vojenské výdaje. Splnění tohoto závazku je však spojeno s propočty, které komplikuje i tak relativní systém cen, rozptýlenost zbrojních zakázek po více než patnácti ministerstvech, a možná také v rozhodující míře to, že nikdo vážně výdaje nepočítal. Nikoliv ze sovětských, ale z amerických publikací se dozvídáme, že jeden z významných sovětských vojenských činitelů za návštěvy v USA uvedl, že ani sám skutečný rozsah vojenských výdajů nezná. Neznáme celkový početní stav našich ozbrojených sil, ačkoliv na Západě, například v USA, se početní stav ozbrojených sil pravidelně oznamuje. Nevíme, jakou část z celkového počtu našich tanků tvoří 10 000 tanků, které se mají zlikvidovat, a kolik tankových divízi zůstane ve východní Evropě poté, až jich odtud bude šest staženo.“ Tyto skutečnosti podle komentátora nejen brání důvěře okolního světa vůči SSSR, ale především brání racionálnímu řízení země, racionální mobilizaci sil a zdrojů k přestavbě.

Tolik Rudé právo z 5. ledna 1989.

Léta se v zemích RVHP předpokládalo, že je to nutný úmysl utajovat přesná čísla o čemkoliv, aby „třídni nepřítel“ nevěděl. Dnes se nám toto utajování všeho do nedávné minulosti mstí. Rozrostlo se tak, že jsme neschopní mít jakoukoli skutečně pravdivou evidenci o čemkoli, čili jsme v totální nevědomosti o skutečném stavu věcí a to kdekoli. Západní země nám proto nevěří. „Nevěří našim údajům a provádějí si vlastní průzkumy a evidenci jak to vlastně v zemích RVHP skutečně vypadá.“ (E. Ševarnadze). Všeobecná neinformovanost, ale i neznalost skutečného stavu věcí vyplývá také z totální nedisciplinovanosti vyplňování různých formulářů a hlášení (tuto nedisciplinovanost ovšem podporuje jejich vlastní ubohost a často i nesmyslnost) vymyšlenými nebo vhodně upravenými údaji tak, aby vyhovovaly prání nadřazených složek. Zpracované výsledky jsou pak přirozeně mylnými podklady pro centrální zpracování, z kterého dále vznikají nereálné plány, scestné směrnice, nesplnitelná usnesení. Dodavatelsko-odběratelské vztahy jsou narušeny, úkoly se neplní, prostojí a dluhy rostou. A to především proto, že informace není strojově objektivní. Člověk ji zkreslil, upravil, znehodnotil. Komplexní hromadné nasazení a plné využití výpočetní techniky, automatizace sběru a vyhodnocování údajů by subjektivní vlivy člověka výrazně omezilo, přesunulo na periferii rozhodovacího procesu. To by se ovšem některým bonapartistům nelíbilo, ale stroj vždy rozhoduje objektivně, nestranně, vědecky přesně. A jedině exaktní řízení společnosti založené na objektivně pravdivé informovanosti je v moderní době možné.

Nasazení výpočetní, automatizační a další moderní techniky v řízení, vyhodnocování, rychlé koordinaci atd. přímo v masovém měřítku do všech hospodářských a řídicích struktur ve všech zemích západního hospodářského systému (EHS) umožnilo mimo jiné výhled sjednocení celé ekonomiky do jednotného tržního systému po roce 1992. S „Nysami“ a „šcoty“ by to nikdy nebylo možné. Takováto nadnárodní, mohutná organizace by nikdy nemohla být rentabilní a bez dokonalého vybavení nejmodernější technikou by se nutně zpátky atomizovala a přijala opět přísná celní, vývozní a další opatření. Bez supermoderní techniky v pokrokové industriální společnosti není tedy možný rychlý koordinovaný vzájemně výhodný postup vpřed.

Amatérská zájmová činnost, stavba nejrůznějších přístrojů a zařízení patří, patří a budou patřit mezi krůčky v tomto směru.

Ing. Jan Klábal

ADAPTER PRO PŘÍJEM TELETEXTU POMOCÍ MIKROPOČÍTAČE

Ing. Lumír Příbyl, Pavel Brychta V. Noska 67, 664 11 Zbýšov

Jedním z nových druhů informačních služeb při současném využití stávajících tras pro přenos informací jsou systémy využívající dosud volných řádků v televizním signálu. Systém, který pro přenos informace digitálním způsobem využívá volných řádků v pulsniřkovém zatemňovacím intervalu, dostal obecný název **TELETEXT**.

Teletextová služba byla poprvé realizována ve Velké Británii v polovině sedmdesátých let u státní společnosti BBC pod názvem Ceefax a u komerční IBA pod názvem Oracle.

Teletextové systémy jsou postupně zdokonalovány a v této souvislosti existuje dělení přenosových systémů na pět tzv. úrovní:

Úroveň 1 — základní abeceda bez diakritických znamének, šest základních barev, relativně hrubá grafika. Tuto úroveň má britská verze teletextu v základním provedení.

Úroveň 2 — abeceda včetně diakritických znamének, pastelové barvy, hladké přechody mezi symboly mozaikového typu. Tato úroveň se částečně využívá při aplikaci britského teletextu pro jiné jazyky.

Úroveň 3 — zavádí se tzv. DRCS (Dynamically Redefinable Character Set) — dynamicky redefinovatelné soubory znaků, tzn. nejprve se z centra nadefinují netypizované znaky a symboly, pak jsou vyvolávány obdobně jako znaky typizované. V této úrovni je možno vytvářet grafiku s vysokou rozlišovací schopností, rozšířit počet barev, používat i obrázkového písma (japonština a čínština).

Úroveň 4 — alfageometrické kódování; definuje zobrazované znaky s použitím pěti základních povelů pro kreslení bodů, čar a ploch. Lze zobrazit obrazy bohaté na podrobnosti, které se kvalitou blíží fotografii. V této úrovni pracuje např. kanadský systém Telidon.

Úroveň 5 — alfafotografická metoda — umožňuje přenos statických obrazů s vysokou rozlišovací schopností. Obraz má zpočátku hrubou strukturu a s postupujícím časem se zjemňuje. Tato metoda je použita při vývoji japonského systému Captain.

V Československu bylo po zvážení technických i ekonomických faktorů rozhodnuto zavést systém WST (World System Teletext — takto je z prestižních a reklamních důvodů nazýván britský systém teletextu). V sousedních zemích se tímto systémem buď už vysílá (NSR, Rakousko, Maďarsko) nebo se vysílání připravuje (Polsko).

Základní technická specifikace systému WST:

1. Data se přenášejí rychlostí 6,9375 Mbit/s v tzv. „neviditelných“ tv řá-

cích (tj. řádcích č. 7 až 22 a 320 až 335, které následují těsně po snímkovém synchronizačním impulsu, ale nejsou zobrazovány). Pokud není vysílán běžný tv signál, lze využít všech tv řádků. Během jednoho tv řádku se přenese informace o celém textovém řádku, tj. číslo magazínu, číslo řádku a 40 znaků, v případě nultého řádku číslo magazínu, číslo řádku, číslo stránky a podstránky, řídicí kódy a 32 znaků.

2. Každá zobrazovaná stránka má 24 textových řádků po 40 znacích. První řádek stránky (tj. řádek č. 0) se nazývá záhlaví. Jsou v něm přenášeny řídicí informace nutné pro zobrazení stránky, dále číslo zobrazené stránky, název informační služby (např. CST-TEXT nebo ORF-TELETEXT), datum a přesný čas.

3. Maximální počet přenášených stránek je 800, každá z těchto stránek může mít až 50 přířazených „podstránek“. Stránky se rozdělují do tzv. magazínů po 100 stránkách. Doba přenosu jednoho magazínu je asi 25 s při využití dvou řádků v pulsniřkovém zatemňovacím intervalu. Při větším počtu magazínů nebo při využití „podstránek“ se doba přenosu celé relace příslušně prodlužuje.

4. Rychlost přenosu je možné zvětšit zvětšením počtu využitých řádků (např. 4 nebo 6 v pulsniřkovém zatemňovacím intervalu).

5. Informace jsou přenášeny sériově pomocí digitálních slov s jednotnou délkou osm bitů. Sedm bitů je informačních, osmý bit je paritní. U adres a důležitých funkcí je použito ochrany samokorigujícím Hammingovým kódem.

6. Systém umožňuje zobrazení textu do aktivního obrazu, např. titulování pořadů pro neslyšící nebo zobrazení tzv. „newsflash“, tj. důležité krátké zprávy, která se na obrazovce objevuje v okamžiku vysílání.

7. Může být použito osm barev pro pozadí znaků a vlastní znak. Dále je možné zobrazovat znaky s dvojnásobnou výškou.

Podrobný popis je uveden v souhrnném přehledu „TELETEXT — popis systému WST úrovně 1.5 na str. 9.

Stav v ČSSR

V Československu do konce roku 1988 probíhalo experimentální vysílání, určené především pro pracovníky redakcí teletextu a pracovníky spojů. Od 1. 1. 1989 je zavedeno pravidelné vysílání. Předpokládá se, že hlavní náplní této nové služby bude alespoň z počátku pomoc národnímu hospodářství, tj. sdělování informací z oblasti průmyslu, zemědělství, dopravy, zdravotnictví apod. Dále bude nabízet programy rozhlasu a televize, divadel, informace o počasí, výukové programy, zpravodajské informace atd.

Specifickým problémem zavedení teletextu v ČSSR je existence dvou spisovných jazyků, resp. i některých jazyků menšinových. Rozšíření souboru znaků tak, aby dokonale zobrazil všechna diakritická znaménka, užívaná v češtině a slovenštině, a dále o znaky užívané v abecedách sousedních zemí, je úzce spojeno s volbou systému teletextu.

Základní tabulka znaků G0, platná pro WST úrovně 1 (viz článek „TELETEXT — popis...“) je navržena pro angličtinu. Při postupném zavádění WST v evropských zemích musel být splněn požadavek na zobrazení specifických národních abeced. Tak vznikly další normalizované národní tabulky, kde jsou na třinácti definovaných pozicích umístěny některé národní znaky.

Čeština spolu se slovenštinou obsahuje značně větší počet zvláštních znaků. Proto musel být systém na úrovni 1 rozšířen alespoň při přenosu alfanumerických znaků na úroveň 2. Tím se však poněkud prodlouží doba přenosu a zvýší se nároky na funkci dekodéru. Úroveň takové soustavy se označuje jako „úroveň 1,5“.

Prakticky se přenos dat na této úrovni provádí tak, že jsou vysílány tzv. neviditelné řádky (pakety), které mají podobný formát jako textové řádky, ale mají číslo vyšší než 24 a nesou doplňující informace pro dekodér znaků.

Pro příjem informací přenášených pomocí teletextu je třeba, aby televizní přijímač byl vybaven speciálním dekodérem.

Teletextové dekodéry bývají zabudovány přímo do televizních přijímačů a jsou sestaveny z jednoúčelových obvodů VLSI, vyvinutých pouze pro tento účel.

Z dosavadního vývoje je zřejmé, že v ČSSR bude otázka součástkové základny pro dekodéry řešena dovozem buď součástek nebo dekodérů, případně celých TV přijímačů. Dekodéry se samozřejmě budou montovat do nových přístrojů, protože podle vyjádření příslušných odborníků dosud vyráběné přijímače nevyhovují koncepčně pro kvalitní příjem teletextu. Z tohoto důvodu, a také proto, že dekodéry sestavené ze speciálních obvodů jsou určeny pro přímé vestavění do televizoru, nebude možné doplňovat stávající tv přijímače těmito dekodéry. V současné době u nás není tedy na čem tyto informace přijímat. Je vhodné zamyslet se nad možností, jak využít stávajících tv přijímačů (i když nejsou zcela vhodné) a případně dalších dostupných přístrojů tak, aby bylo možné teletext dekódovat a zobrazovat na stávajících televizorech i za cenu snížených kvalitativních požadavků.

Situace je podobná té, která vznikla v ČSSR při zavádění II. tv programu. Tehdy byla řešena pomocí konvertorů, tj. přístrojů, které umožnily příjem druhého programu i na starších tv přijímačích. Toto řešení bylo sice nepraktické (nutnost přepínání a nastavování dvou ladicích prvků), ale přesto umožnilo příjem bez nutnosti zásahu do zapojení televizoru.

Navrhované řešení

Koncepce použití „vnějšího“ dekodéru teletextu je založena na těchto faktech:

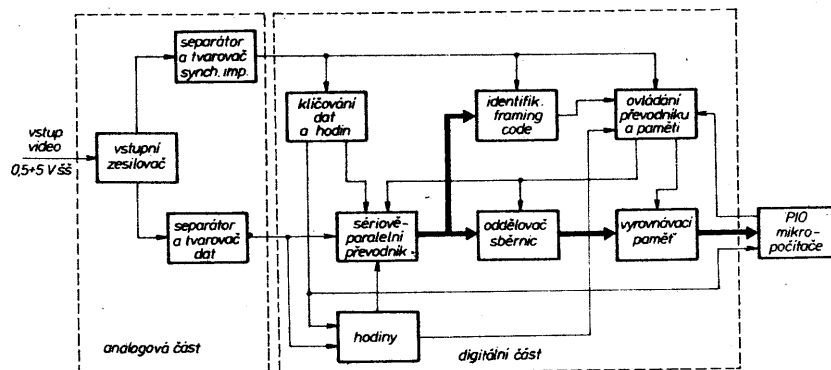
1. Přenosové charakteristiky moderních barevných televizorů zaručují nekreslený přenos barvonosných kmitočtů (střední kmitočet kolem 4,4 MHz). Dále bývají vybaveny vstupem a výstupem videosignálu se standardní úrovní. Tento videosignál samozřejmě obsahuje i řádky nesoucí teletextová data. (Praktickými zkouškami bylo zjištěno, že pro příjem v místech kvalitního signálu vyhoví i přenosné černobílé televizory s doplněným videovýstupem.)
2. Stále více se rozšiřují osobní počítače, které umožňují zpracovat vhodné upravenou informaci a zobrazit ji na obrazovce televizoru. Barevné zobrazení, používané i u jednodušších počítačů, odpovídá částečně upraveným požadavkům na zobrazení teletextu.

Na těchto skutečnostech je založena myšlenka využít osobního počítače pro dekódování a zobrazování teletextu.

Samotný mikropočítač samozřejmě není schopen přímo zpracovat videosignál z televizoru. Je třeba vytvořit vhodný mezičlánek (adaptér), který umožní vybrat data, vysílaná v příslušných tv řádcích, převést je ze sériového do paralelního tvaru a uschovat je na potřebnou dobu ve vyrovnávací paměti. Z této paměti si je pak v daném okamžiku mikropočítač přesune do své operační paměti, vhodným způsobem zpracuje, a zobrazí na obrazovce.

Z uvedeného způsobu dekódování vyplývají některá omezení a nevýhody proti standardnímu způsobu dekódování:

1. Základní sestava pro příjem teletextu je složena z tv přijímače, adaptéru,



Obr. 1. Blokové schéma adaptéru

mikropočítače a monitoru pro mikropočítač. V praxi to znamená použití dvou tv přijímačů.

2. Při požadavku na použití pouze jednoho televizoru je nutný buď zásah do přijímače nebo by bylo možné při příjmu tv signálu obsahujícího teletext požadovanou stránku načíst do paměti mikropočítače, pak přepnout tv přijímač jako monitor a stránku zobrazit. V druhém případě by tedy během výběru a zpracování dat byl na obrazovce televizoru program přijímaného vysílače (u standardního dekodéru je stále zobrazována poslední načtená stránka). Dále by byla nutná zvuková signalizace načtení stránky. V případě, že zvolená stránka má rotující podstránky, je tento způsob velmi nepraktický.
3. Vzhledem ke způsobu zobrazování mikropočítačů pomocí „okna“ na obrazovce (ZX-Spectrum, Sord, Sharp) je zobrazená stránka rozměrově menší; to má za následek zhoršenou jakost obrazu.
4. Není možné vkopírovat zobrazované informace do tv obrazu, tj. nelze např. titulkovat vysílané pořady.
5. Každý mikropočítač je zdrojem poměrně intenzivního rušení. Také sám adaptér ruší, i když podstatně méně než mikropočítač. Při použití uvedeného způsobu příjmu teletextu je vlastně mikropočítač přes televizor připojen k anténě. Z toho plyne především nutnost zkontrolovat možnost rušení okolních televizních a rozhlasových přijímačů zvláště v případě, že se jedná o příjem na společnou anténu. Dále je nutné počítat s tím, že příjem teletextu bude zcela znemožněn, pokud přijímaný vysílač bude v I. nebo II. pásmu, a bude obtížný při příjmu vysílačů v dolním konci III. pásma (kanál č. 5 CCIR-B a č. 6 CCIR-D). Tento problém se netýká jenom příjmu zahraničních vysílačů. Čs. teletext je sice vysílán pouze v síti druhého programu (tj. pásmo IV. a V.), ale ve společných anténách bývá přijímaný signál převáděn na některý kanál v pásmu I. a II. a jeho využití pro příjem teletextu tímto způsobem je znemožněno. Řešením je pak pouze individuální příjem druhého programu.

Dekódování teletextu mikropočítačem naopak umožňuje snadné zpracování přijatých dat, jejich přenos do jiné informační sítě a další zpracování. Dále je možné úpravou programového vybavení modifikovat zpracování informací podle požadavků uživatele. Relativně velká operační paměť mikropočítače

umožňuje i takový způsob dekódování, kdy je několik požadovaných stránek nejprve načteno a zpracováno a pak jsou postupně v libovolném pořadí zobrazovány bez jakýchkoli ztrátových časů.

Koncepce adaptéru pro dekódování teletextu mikropočítačem

Adaptér se skládá ze dvou základních částí — analogové a digitální, viz blokové schéma na obr. 1.

V analogové části je videosignál nejprve zesílen ve vstupním zesilovači, pak jsou z něho odděleny synchronizační impulsy a vlastní data. Tyto dva signály jsou dále zpracovávány v digitální části adaptéru. Ze synchronizačních impulsů jsou odvozeny klíčovací impulsy pro řízení výběru dat a sériově-paralelního převodu. Data, převedená do paralelního tvaru, jsou přiváděna do identifikačního obvodu, který reaguje na identifikační slovo teletextu (framing code). V případě, že je toto identifikační slovo nalezeno, zapisí se následující data z příslušného řádku do vyrovnávací paměti. V aktivní době pulsů, tj. v době, kdy je na obrazovce zobrazován obraz vysílaný vysílačem, pak mikropočítač řídí přesun dat z vyrovnávací paměti do vlastní operační paměti a dále je zpracovává podle zadanych požadavků.

Ve skutečnosti nejsou jednotlivé funkční bloky ostře odděleny, ale naopak, z důvodů co nejjednoduššího zapojení se navzájem prolínají.

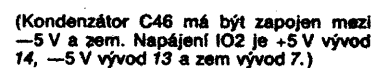
Popis zapojení a funkce jednotlivých obvodů

Zapojení adaptéru a napájecího zdroje je uvedeno na obr. 2. a 3.

Analogová část vlastního adaptéru je řešena klasickým způsobem pomocí diskretních součástek, digitální část využívá obvodů TTL SSI, MSI a LSI.

Tranzistory T1, T2, T3 a T3' tvoří vstupní zesilovač, jehož zesílení lze nastavit proměnným rezistorem R5. Předpokládá se, že velikost mezivrochového napětí vstupního signálu je v rozsahu 0,5 až 5 V. Tranzistory T3 a T3' vytváří oddělené výstupy videosignálu pro separátor a tvarovač dat a separátor a tvarovač synchronizačních impulsů.

Z emitoru tranzistoru T3' je signál přiveden do obnovitele stejnosměrné



složky tvořeného kondenzátorem C8, rezistorem R14 a diodou D1. Ten upíná temena synchronizačních impulsů na úroveň asi -1 V . Vlastní obrazový signál prochází dolní propustí, která je tvořena rezistorem R15 a kondenzátorem C9. Takto upravený signál je přiveden na neinverující vstup komparátoru synchronizačních impulsů (1/2 IO2). Inverující vstup je spojen s jezdcem nastavitelného rezistoru R26. Tímto trimrem se nastavuje vzorkovací úroveň. Na výstupu komparátoru pak získáme synchronizační směr obsahující řádkové i snímkové synchronizační impulsy (H). V obvodu složeném z IO3, R18, R19, C19 jsou z této směsi odděleny snímkové synchronizační impulsy (V, V).

Od snímkového synchronizačního impulsu (V) je odvozen pomocí dvou monostabilních klopných obvodů (IO4) signál GO, který vymezuje dobu, po kterou se předpokládá příchod řádků obsahujících teletext (tj. řádky č. 7 až 22 a 320 až 335). Přesné časové umístění signálu GO lze nastavit pomocí R21 a R23.

Z emitoru tranzistoru T3 je přes kondenzátor C4 přiváděn videosignál přímo na neinverující vstup komparátoru (2/2 IO2). Obvod C5, R9, 2/4 IO1, R10, C49 a T4 tvoří spolu s C4 obnovitel stejnosměrné složky. Upnutím na úroveň černé se odstraňuje vliv kapacitní vazby na posun úrovní videosignálu při kolísání středního jasu. Vzorkovací úroveň, přiváděná na inverující vstup komparátoru z R13, musí

být nezávisle na velikosti přiváděného videosignálu vždy uprostřed mezi úrovněmi log. 0 a log. 1. Toho dosáhneme tak, že napájíme rezistor R13 ze špičkového usměrňovače T5, R11 a C6 přes tranzistor T6, který slouží pro posun úrovně napětí. Napětí na emitoru T6 se při změnách velikosti přiváděného signálu mění souhlasně s úrovní odpovídající log. 1. Tím je zajištěno automatické udržování vzorkovací úrovně na požadované hodnotě. Na výstupu komparátoru dostaneme signál DATA, který obsahuje teletextová data v sériovém tvaru. Kondenzátory C16 a C16' slouží k „doladění“ (tvarování) signálu DATA.

Ze signálu DATA jsou v obvodu tvořeném IO5, rezistory R24, R25, R26, R27 a kondenzátory C17 a C18 získány synchronizační impulsy pro generátor hodinového kmitočtu 6,9375 MHz. Generátor využívá jedno hradlo IO6 spolu s C21, C22 a L1 jako oscilátor a zbývající hradla slouží jako invertory. Na výstupu těchto invertorů získáme signál CLK a CLK, který je nutný pro sériově-paralelní převodník. Generátor je klíčován signálem GO, tzn. že kmitá pouze v době předpokládaného příchodu teletextových dat.

Sériově-paralelní převodník je realizován pomocí posuvného registru IO7, obvodu IO8, zapojeného jako osmičkový čítač, a mezipaměti (budiče sběrnice) IO11. Posuvný registr je klíčován signálem GO. Na vstup posuvného registru jsou přiváděna data v sériovém

tvaru, posuv dat je prováděn v rytmu hodinového kmitočtu CLK. Současně je signál CLK přiváděn do osmičkového čítače, který vytváří zapisovací impulsy (STB1) pro zápis jednoho bajtu do mezipaměti IO11.

Obvody IO9, IO10 a 2/4 IO1 tvoří identifikační obvod, který reaguje na přítomnost tzv. rámcového kódu na začátku teletextového tv řádku. Jestliže se na výstupu posuvného registru objeví tento kód, pak se na výstupu IO10 (vývod č. 8) vytvoří impuls (FR). Pokud se tento impuls vyskytne na začátku řádku (kontrola pomocí 2/4 IO1), je pak z tohoto signálu a ze signálu H pomocí klopného obvodu (2/4 IO14) vytvořen signál WE a WE, který slouží k ovládání mezipaměti (IO11) a vyrovnávací paměti (IO19, IO20). Dále je od signálu WE pomocí 3/4 IO12 odvozen synchronizační impuls pro osmičkový čítač (IO8).

Vyrovnávací paměť 1kB tvoří obvody IO19 a IO20. Pro adresování této paměti slouží tři šestnáctkové čítače IO16, IO17 a IO18. Inkrementování těchto čítačů je prováděno při zápisu impulsy (STB1) získanými v osmičkovém čítači (IO8), které slouží současně pro zápis do mezipaměti (IO11), při čtení pak impulsy přicházejícími přes IO15 z mikropočítače (STB2). Přesný okamžik přepisu jednoho bajtu z mezipaměti do vyrovnávací paměti (signál CS MEM) je odvozen pomocí IO13 v závislosti na stavu osmičkového čítače (IO8).

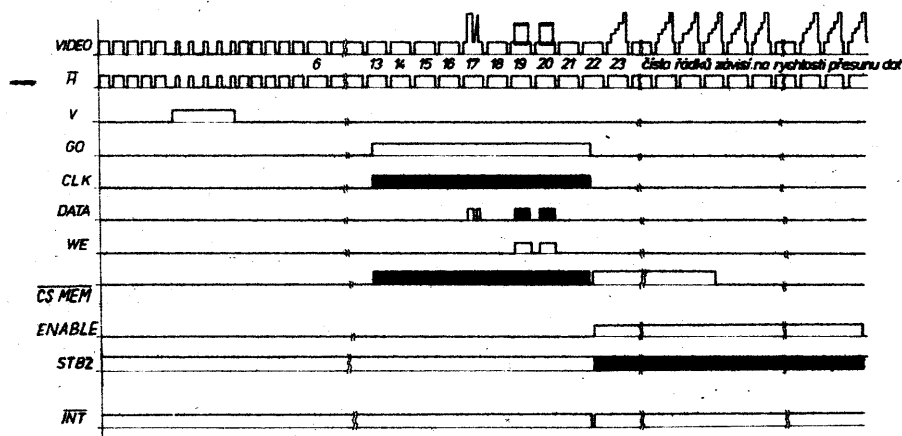
Protože vyrovnávací paměť není nulovaná, je třeba při čtení vymaskovat ty bajty, které nebyly při předchozím zápisu přepsány. To je provedeno pomocí IO22. Čítací cyklus je zkrácen, je využíváno pouze 448 bajtů vyrovnávací paměti. Po příchodu snímkového synchronizačního impulsu je čítač vynulován, během aktivního signálu GO a WE se inkrementuje a zastaví se na určité adrese. Od této adresy pokračuje po skončení aktivního signálu GO v inkrementování mikropočítače, paměť je však odpojena od sběrnice. Teprve až dojde k „přetečení“ čítače a jeho vynulování, je paměť připojena ke sběrnici. Jelikož mikropočítač čte pouze 448 bajtů, je zaručeno, že přečte pouze skutečně požadované bajty, ostatní budou zamaskovány hodnotou FF (hex).

První monostabilní klopný obvod v IO21 slouží k prodloužení signálu WE pro signalizaci přítomnosti teletextového signálu. Druhý monostabilní klopný obvod vytváří po skončení signálu GO impuls nutný pro činnost maskovacího obvodu (signál INT).

Průběhy důležitých signálů během jednoho půlsnímků a během příchodu teletextového datového řádku jsou rozkresleny na obr. 4 a 5.

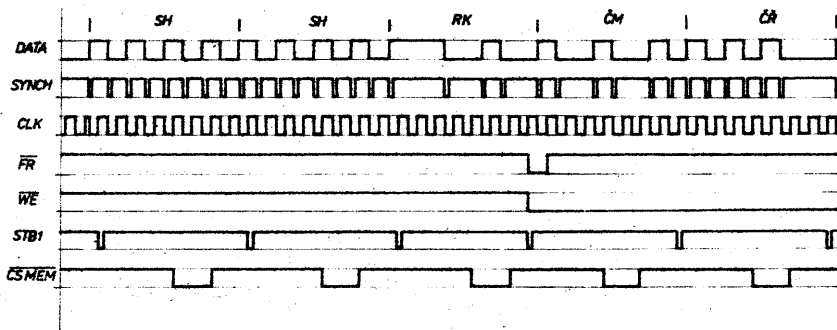
Mikropočítač je k adaptéru připojen přes obvod typu 8255A. Signály CS PIO a RD PIO nejsou připojeny přes porty, ale jsou odebírány přímo z obvodu 8255A (vývod č. 6 a 5).

Z adaptéru do mikropočítače jsou přivedeny signály D0 až D7 a GO. Z mikropočítače do adaptéru jsou přivedeny signály ENABLE, CS PIO a RD PIO.



Pozn.: Signál DATA je citlivý na připojení osciloskopu. Lze si pomoci tak, že snímáme signál na vývodu č. 3 posuvného registru IO7.

Obr. 4. Průběhy hlavních signálů během začátku jednoho půlsnímků



Obr. 5. Průběhy hlavních signálů během příchodu teletextového řádku (průběhy jsou idealizované, skutečné vzájemné umístění signálů závisí na zpoždění použitých obvodů)

Nastavení adaptéru

Na vstup adaptéru je třeba přivést videosignál s mezivrcholovým napětím 0,5 až 5 V viz část „Podmínky dobrého příjmu“. Protože zatím nelze nastavit televizor podle adaptéru, naladíme jej na co nejostřejší obraz.

Zesílení vstupního zesilovače se nastaví rezistorem R5 tak, aby na rezistorech R7 a R8 bylo mezivrcholové napětí asi 3 až 5 V. Při nižší kvalitě signálu je třeba nastavit větší amplitudu.

Pak nastavíme rezistorem R16 pomocí osciloskopu vzorkovací úroveň tak, aby na výstupu komparátoru synchronizačních impulsů byly tyto impulsy stabilní, ale přitom nedocházelo ke vzniku falešných impulsů. Zkontrolujeme průběhy signálů H, V a V, které musí odpovídat průběhům na obr. 4. Pak zkontrolujeme a nastavíme proměnnými rezistory R21 a R23 signál GO a GÖ. Při nastavování se orientujeme podle měřícího řádku. Pro tyto práce je vhodný dvoukanalový osciloskop nebo alespoň osciloskop s možností externí synchronizace. Při nastavování signálu GO osciloskop synchronizujeme signálem V. Při nastavování je třeba vzít v úvahu, že mohou být zobrazovány sudé i liché půlsnímků, které jsou navzájem o půl řádku posunuty. Z toho plyne, že všechny signály spjaté s řádkovými synchronizačními impulsy se mohou zobrazovat ve dvou pozicích. Naopak signál GO, který je odvozen od snímkových synchronizačních impulsů, musí být naprosto stabilní a musí být nastaven s jistým přesahem tak, aby překrýval výskyt teletextových řádků v sudém i lichém půlsnímku. Je vhodné nastavit šířku aktivního signálu GO podle skutečně vysílaných teletextových řádků. V případě příjmu více různých systémů se nastaví šířka GO podle nejrozsáhlejšího z nich.

Pro přenos teletextu mohou být obecně použity řádky č. 7 až 22 (resp. 320 až 335). Čs. televize využívá zatím řádky č. 19, 20 (332, 333). Měřicí řádek má č. 17 (330).

Oscilátor hodin předem naladíme (s kontrolou čítačem) tak, aby volně kmital na kmitočtu asi 7 MHz. Pro nastavení je třeba odpojit signál GO od vstupu č. 3 obvodu IO6, nepřipojovat videosignál, aby nedocházelo k tvarování synchronizačních impulsů (signál SYNCH) a čítač připojit na vývod č. 8 nebo 12 obvodu IO6.

Jestliže máme předběžně nastavený oscilátor hodin, připojíme osciloskop na signál WE (osciloskop synchronizujeme signálem GO) a proměnným rezistorem R13 se snažíme nastavit takovou vzorkovací úroveň pro komparátor dat, aby průběh signálu WE odpovídal průběhu na obr. 4 a byl stabilní. Můžeme si pomoci i jemným doladěním oscilátoru a doladěním tv přijímače, ze kterého získáváme videosignál (viz část „Podmínky dobrého příjmu“). Počet impulsů signálu WE musí odpovídat počtu přítomných teletextových řádků. Dále se mohou nepravdivě vyskytnout i falešné impulsy odpovídající řádkům se synchronizačními impulsy soustavy SECAM nebo měřicímu řádku.

Nakonec nastavíme pomocí proměnného rezistoru R32 zpoždění signálu WE

na dobu o něco delší než je doba trvání jednoho půlsnímku, tzn., že při přítomnosti teletextových dat bude na vývodu č. 5 obvodu IO21 trvale log. 1. Dioda D5 pak indikuje přítomnost teletextových řádků.

Tím je předběžné nastavení ukončeno. Je možné ještě zkontrolovat, zda v průběhu aktivního signálu WE čítají čítače IO8 a IO16, IO17 a IO18, a zda po skončení impulsu GO je v IO21 vytvářen impuls (signál INT; viz obr. 4). Dále je možné vhodným osciloskopem nebo log. analyzátozem zkontrolovat průběhy signálu podle obr. 5. Při použití dobrých součástek by však mělo být vše v pořádku.

Další nastavení se provádí pomocí mikropočítače. Pro ověření činnosti adaptéru slouží v příloze 1 uvedený program v BASICu a rutina ve strojovém kódu. Jsou určeny pro mikropočítač ZX Spectrum, lze je však snadno upravit i pro jiné počítače. Při úpravě je především nutné předadresování vstupních a výstupních portů obvodu 8255A podle daného počítače.

Připojení adaptéru a interfejsu s obvodem 8255A je uvedeno v tabulce obr. 6.

| Vývod obvodu | Signál | Signál | Vývod adaptéru |
|--------------|--------|--------|----------------|
| 8255A | | | |
| 4 | PA0 | D0 | 4 |
| 3 | PA1 | D1 | 6 |
| 2 | PA2 | D2 | 8 |
| 1 | PA3 | D3 | 10 |
| 40 | PA4 | D4 | 12 |
| 39 | PA5 | D5 | 14 |
| 38 | PA6 | D6 | 16 |
| 37 | PA7 | D7 | 18 |
| 14 | PC0 | GÖ | 19 |
| 10 | PC7 | ENABLE | 20 |
| 5 | RD | RD PIO | 27 |
| 6 | CS | CS PIO | 28 |
| 7 | — | — | 30 |

Obr. 6. Připojení adaptéru s mikropočítačem pomocí obvodu 8255

Rutina ve strojovém kódu jednorázově přečte obsah vyrovnávací paměti a ukládá jej do operační paměti od adresy 8032 (hex). Přečte 512 bajtů, tzn. celou vyrovnávací paměť (448 bajtů) a pak dalších 64 bajtů, které už nejsou maskovány. Přečtené údaje se pak zobrazí pomocí programu napsaného v BASICu.

Po spuštění programu a pak po každém stisknutí libovolné klávesy je na obrazovku jednorázově vypsan obsah všech 448 bajtů vyrovnávací paměti. Nejprve jsou vypisovány zamaskované bajty, na konci jsou pak vypsané teletextové řádky včetně tří bajtů obsahujících rámecový kód, číslo magazínu a číslo řádku. Zamaskované bajty jsou na Spectru zobrazovány jako (c), rámecový kód jako apostrof. Za rámecovým kódem následují dva adresovací bajty, pak dalších 40 znakových bajtů teletextového řádku a na konci jsou dva nebo tři nulové bajty. Protože většina kódů pro přenos alfanumerických teletextových znaků se shoduje s kódem ISO-7 (ASCII), měly by být některé řádky čitelné (nebo alespoň jejich začátek). Měla by se zobrazit i záhlaví některých stránek. Řídící znaky ze sloupce 0 a 1 tabulky G0 se zobrazí inverzně. Řádky, které přenášejí grafické znaky nebo doplňkové informace, se

samozřejmě zobrazí pomocí ASCII znaků tak, že nebudou dávat smysl. Dále zkontrolujeme, zda první platný řádek ve vyrovnávací paměti začíná rámecovým kódem (apostrof). Pokud tomu tak není, lze provést následující úpravu zapojení: vývod č. 5 IO14 odpojíme od vývodu č. 2 IO9 a připojíme jej k vývodu č. 4 IO9.

Pokud se budou zobrazovat stále pouze nečitelné řádky, je třeba hledat chybu v adaptéru. Ze zobrazovaných znaků je možné např. usoudit, zda se přenáší správně všechny bity D0 až D7, zda funguje maskování paměti atd.

Konečné nastavení provádíme s programem TELETEXT. Pokud se při použití testovacího programu podařilo načítat alespoň začátky některých řádků, mělo by být možné po spuštění programu TELETEXT načíst teletextové testovací stránky (v českém vysílání magazín č. 8, ve slovenském č. 3).

Nejprve opakovaně načítáme stránku s názvem ZKOUSKA SYNCHRONIZACE DEKODÉRU a nastavíme přesně kmitočet hodin, případně úroveň vzorkování dat. Pak opakovaně načítáme stránku OPTIMÁLNÍ SYNCHRONIZACE DEKODÉRU a pokud je třeba, dostavíme úroveň vzorkování dat, případně ještě jemně doladíme kmitočet hodin. Nakonec načteme stránku ZKUŠEBNÍ OBRAZEC TELETEXT a zkontrolujeme zobrazování všech znaků, módů zobrazení a činnost všech funkcí.

Pokud je vše v pořádku, můžeme se pokusit zmenšovat zesílení vstupního zesilovače (R5) a přitom jemně doladit nastavení komparátorů (R26, R13). Čím kvalitnější je signál, tím menší zesílení můžeme nastavit. (Zpracováním signálu s nízkou úrovní se dále sníží nepříznivý vliv změn středního jasu a kolísání úrovně videosignálu na správnost dekódování.) Optimální amplituda signálu na rezistorech R7 a R8 je asi 3 V, záleží však na zdroji signálu. V některých případech může dávat lepší výsledky i amplituda 5 V.

Při dálkovém příjmu můžeme dále nahradit kondenzátory C16 a C16' otočným kondenzátorem o kapacitě asi 500 pF a pokusit se při opakovaném načítání zkušebních stránek nastavit příjem s co nejmenším počtem chyb.

Konstrukční řešení

Celý adaptér včetně zdroje byl postaven na univerzální desce. Spojby byly vedeny vodičem se samopájitelnou izolací. Rozložení součástek není kritické s výjimkou vlastního sériově-paralelního převodníku a generátoru hodin. Tyto obvody, tj. IO5, IO6, IO7 a IO10 je vhodné umístit co nejbližší k sobě. Dále je třeba napájet tyto obvody z jednoho uzlu a tento uzel blokovat tantalovým kondenzátorem 4,7 µF.

U všech IO byl do napájení zapojen blokovací kondenzátor 68 nF (100 nF), u IO2 byly připojeny na kladnou i zápornou větev napájení.

Jako L1 byly používány neupravované cívky ze zvukových mf modulů čs. televizorů. Přesný počet závitů není kritický, hrubě je možné oscilátor doladit změnou kapacity kondenzátorů C21 a C22. Poměr C22/C21 by měl být přibližně 1,5.

Diody D2 a D3 byly použity z toho důvodu, že při experimentálním ověřování byl generovaný signál GO stabilnější, než při použití zapojení bez

Příloha 1.

Testovací program v BASiCu a rutina pro přečtení obsahu vyrovnávací paměti

```

1 REM LOAD **CODE 32768,100
5 RANDOMIZE USR 32768
10 RANDOMIZE USR 32791
20 LET i=32818
35 CLS : GO SUB 1000
90 IF INKEY$="" THEN GO TO 90
100 GO TO 10
999 STOP
1000 FOR j=i TO i+447
1010 LET a=PEEK j
1012 IF (a=39) AND (j<=i+446) THEN PRINT
1018 IF a>=128 THEN LET a=a-128
1020 IF a>=32 THEN PRINT CHR$(a);
1030 IF a<32 THEN PRINT "O: INK 7: PRINT CHR$(a+64);: PAPER 7: INK 0
1040 NEXT j
1050 RETURN

```

```

.z80 ; use z80 cpu
.phase 8000h ; origin for ZX-Spectrum
; teletext test program
; i/o port assignments
; pa - data from teletext
; pc0 - /go signal from teletext
; pc7 - enable signal to teletext
; program equates

```

```

001F pa equ 1fh ; pio port a
005F pc equ 5fh ; pio port b
007F cr equ 7fh ; pio control register
0093 cw equ 10010011b ; control word for pio
000F enb1 equ 00001111b ; enable=1 command
000E enb0 equ 00001110b ; enable=0 command

```

; startup procedure - ppi initialization

```

init:
ld a,cw ; send control word to ppi
out (cr),a
ld a,enb10 ; and reset enable signal
out (cr),a
jr main ; jump to main program

```

; waiting for go signal

```

waitgo:
in a,(pc) ; check go bit
bit 0,a ; active in low
jr nz,waitgo ; go not found

```

```

waitgol:
in a,(pc) ; check go bit again
bit 0,a ; now active in high
jr z,waitgol ; go active - wait
ret ; return to caller

```

; get the teletext lines to buffer

```

main:
di ; disable interrupts
call waitgo ; waiting for go signal is finished
ld a,enb1 ; set enable signal to 1
out (cr),a
ld bc,pa ; port base address+256* # of bytes (=256)
ld hl,buffer ; store area for data
inir ; get data from decoder
ld bc,pa ; port base address+256* # of bytes
inir ; get data from decoder
ld a,enb10 ; reset enable signal
out (cr),a
ei ; enable interrupts
ret ; return to basic
; buffer area
buffer:dfs 512
end

```

Macros:

Symbols:

| | | |
|-------------|--------------|-----------|
| 8032 BUFFER | 007F CR | 0093 CW |
| 000E ENB0 | 000F ENB1 | 8000 INIT |
| 8017 MAIN | 001F PA | 005F PC |
| 800A WAITGO | 8010 WAITGO1 | |

No Fatal error(s)

těchto diod. Jako C13 a C15 je vhodné použít stabilní kondenzátory, není to však nutná podmínka.

Předepsané typy IO řady 74ALS, 74S a 74 musí být dodrženy u IO3, IO6 a IO15. Na místě IO5 je možné použít typ 7403. U ostatních obvodů lze použít řadu 74S i 74. Při použití typů 74 současně na pozicích IO9, IO10, IO1 a IO14 může dojít vzhledem k velkému zpoždění k chybné funkci sériově-paralelního převodníku. V tomto případě je možné přímo propojit vývod č. 8 IO10 s vývodem č. 10 IO14. Pak je adaptér schopen pracovat spolehlivě i s těmito obvody, ale za určitých okolností může být dekodér náchylnější k výskytu chybných řádků.

Adaptér je s mikropočítačem propojen třináctižilovým plochým vodičem délky asi 35 cm. Videosignál z tv přijímače je přiveden stíněným nf kabelem délky asi 1 m.

Podmínky dobrého příjmu

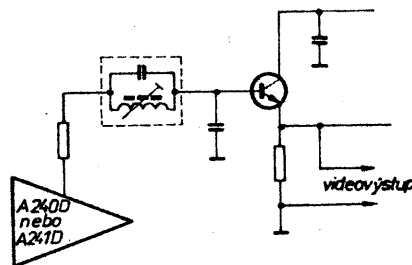
Pro dobrý příjem teletextu je nutné splnění těchto dvou základních předpokladů:

- Vhodný televizní přijímač, ze kterého získáváme videosignál pro zpracování v adaptéru.
- Dokonalý příjem příslušného tv vysílače.

Pod označením vhodný přijímač je myšlen především přijímač pro barevný obraz. V některých případech lze použít i dobře seřazený černobílý přijímač, pokud má vstupní díl a obrazovou mezifrekvenci zapojeny obdobně jako barevné přijímače. To znamená, že vhodné jsou ty čs. přijímače, které mají obrazovou mezifrekvenci osazenou integrovaným obvodem A240D nebo A241D. Zahraniční televizory, zapojené obdobně, budou patrně také vhodné, nebyly však autory vyzkoušeny. Rozhodně se nehodí starší typy televizorů, které mají obrazovou mezifrekvenci osazenou tranzistory nebo dokonce elektronkami.

Pokud je napájení televizoru řešeno bez galvanického oddělení od sítě, je nutný oddělovací transformátor.

Videosignál lze získat buď ze standardního videovýstupu některých tv přijímačů (Oravan, Mánes, Aleš) nebo je nutné výstup do tv přijímače doplnit. Pokud není zaručen dokonalý příjem nebo jde o dálkový příjem zahraničních vysílačů, pracujících v normě CCIR-B a CCIR-G, je zvláštní videovýstup nutný vždy.



Obr. 7. Zapojení videovýstupu

Jednoduché, ale osvědčené zapojení videovýstupu je na obr. 7. Toto zapojení bylo úspěšně vyzkoušeno u televizorů Oravan, Merkur a Pluto. Pro vyvede-

ni signálu na zadní panel televizoru je použito nestíněné lanko.

Pro dobrý příjem je dále nutné přesné naladění tv přijímače. Hrubě naladíme přijímač podle svitu diody D5, jemně doladíme podle zobrazovaného času (číslce nesmí vypadávat) a pak při opakovaném zobrazení některé zkušební stránky. Optimální naladění se nemusí shodovat s optimálním naladěním obrazu. Pokud televizor nemá vyvedeno doladování AFC, je v některých případech nutné AFC vypnout (většinou otevřením dvířek předvolby nebo tlačítkem).

Další podmínkou pro příjem teletextu je, stejně jako v případě standardních dekodérů, dokonalý příjem příslušného tv vysílače. Příjem musí být bez duchů, barevný obraz nesmí sněžit a nesmí obsahovat rušivé moaré. V případě snížených nároků je možné připustit i částečně zhoršený signál, možnost příjmu je třeba ověřit. V tomto případě je však nezbytné nutné doplnit do televizoru zvláštní videovýstup.

Pro informaci jsou uvedeny dva případy, kde bylo dosaženo vyhovujícího příjmu teletextu i za zhoršených podmínek:

- a) sněžení barevného obrazu na hranici pozorovatelnosti, slabé moaré, obraz ostrý bez „duchů“;
- b) barevný obraz bez pozorovatelného sněžení, zřetelné moaré, obraz mírně znehodnocený „duchy“.

V těchto případech mohou čas od času vypadnout některé znaky nebo i celé řádky.

Silné zarušení obrazu „duchy“ může způsobovat systematické chyby v zobrazování. Na určité stránce např. dochází ke stále stejné chybě, na jiných stránkách se chyby nemusí projevit.

Silné sněžení a silné moaré v obraze většinou úplně znemožní příjem.

Systém je principiálně nejcitlivější na impulsní rušení (komutátorové motorky atp.). Toto rušení způsobuje vypaďování znaků, řádků, skupin řádků a dokonce i celých stránek. Způsobuje i načtení falešných stránek, případně může zcela znemožnit příjem. I rušení, které v obraze není takřka postřehnutelné, se projevuje tímto způsobem. Většinou jde o rušení širokopásmové, lze je tedy identifikovat tak, že televizor přepneme na některý obsazený kanál I. a II. pásma případně některý nižší kanál III. pásma, kde se pak toto rušení projevuje výrazněji. Jediným řešením je odstranění zdroje rušení. Je možné pokusit se ještě o příjem vhodného vysílače na některém kanálu v V. pásmu, kde je vliv rušení většinou menší.

Při příjmu může samozřejmě rušit i vlastní mikro počítač (jak bylo popsáno v části „Navrhované řešení“).

Závěr

Během přípravy článku došlo k následujícím změnám:

- a) Byla vytvořena nová verze programu TELETEX (verze 2.4), která odstraňuje některé nedostatky verze 1.9 a je rozšířena o další funkce.
- b) Pro dosažení kompatibility ZX Spectra a Dodaktiku Gama je nutné pro program TELETEX v. 2.4 připojit signál GO na bit PC2 (původně PC0), tj. na vývod 16 (původně na 14) obvodu 8255A. Pro ostatní počítače platí tabulka na obr. 6 beze změn.
- c) Československá televize vysílá od 9. 1. 1989 teletext v řádcích č. 7, 8,

19, 20 (320, 321, 332, 333). Aby bylo možné nastavit potřebnou šířku signálu GO, je nutné zkratovat rezistor R20 a případně ke kondenzátoru C15 připojit paralelně kondenzátor s kapacitou 33 nF. Ostatní součástky zůstávají beze změn.

- d) Bylo zjištěno, že zesilovače STA mohou omezit tv signál natolik, že i při zdánlivě perfektním obraze je příjem teletextu vyloučen.

Seznam součástek

Integrované obvody:

| | |
|-------------|---------------------|
| IO2 | UCY75107PC |
| IO4, IO21 | UCY74123N |
| IO11 | MHB8282 |
| IO7 | MH74164, MH74164S |
| IO19, IO20 | MHB2114 |
| IO16, IO17, | |
| IO18 | MH7493A |
| IO8 | MH7490A |
| IO15 | MH74ALS00 |
| IO1, IO12, | |
| IO13, IO14 | MH74ALS00, MH74S00, |
| | MH7400 |
| IO3 | MH74S00 |
| IO6 | MH7410 |
| IO22 | MH74ALS10, MH74S10, |
| | MH7410 |
| IO5 | MH74S03, MH7403 |
| IO10 | MH74ALS30, MH74S30, |
| | MH7430 |
| IO9 | MH74ALS04, MH74S04, |
| | MH7404 |
| IO101 | MA7805 |

Tranzistory:

| | |
|-------------|----------------------|
| T1, T3, T3' | KC509 (KC508) |
| T4, T5 | KC507 |
| T7 | BC179 (BC178) |
| T2, T6 | KF517 (KFY16, KFY18) |
| T101 | |

Diody:

| | |
|-----------|-----------|
| D1—D4 | KA206 |
| D101—D104 | KY132/80 |
| D105 | KZ260/5V6 |
| D5 | LQ1732 |
| D106 | LQ1432 |

Keramické kondenzátory:

| | |
|---------------|--------------------------|
| C17, C18 | 47/TK754, 774, 794 |
| C23 | 56/TK754, 774, 794 |
| C9, C16' | 150/TK794, 754, 774 |
| C16 | 220/TK794, 754, 774 |
| C24 | 470/TK794, 774 |
| C6 | 820/TK724, 794 |
| C10, C49 | 1n0/TK744, 724, 794 |
| C7 | 4n7/TK783, 764, 744, 764 |
| C5, C14 | 10n/TK783, 764 |
| C11, C48, | |
| C12 | 22n/TK783, 764 |
| C1', C8, C19, | |
| C20, C27—C46, | |
| C103, C104 | 100n/TK783, 782 |

Svítkové kondenzátory:

| | |
|-----|-----------------------------|
| C13 | 68n/TC215—219 |
| C15 | 100n/TC215—219 |
| C21 | 470/TGL5155—A470/ /10/63 |
| C22 | 680/TGL5155—A680/ /10/63 |

Elektrolyt. kondenzátory (tantal.):

| | |
|------------|-----------------|
| C4, C25, | |
| C26 | 4 μF/TE131—135 |
| C1, C2, C3 | 22 μF/TE131—135 |

Elektrolyt. kondenzátory

| | |
|------------|---------------|
| C101, C102 | 1G/TE 984 |
| C105 | 100 μF/TF 009 |

Rezistory TR 212:

| | |
|----------------|-----|
| R6 | 220 |
| R4, R33, | |
| R102 | 270 |
| R9, R18, | |
| R19 | 390 |
| R24, R25, R26, | |
| R27, R29, | |
| R36 | 470 |
| R7, R8 | 1K0 |

R3, R10, R12,

R28, R31,

R34, R35

R17

R15,

R37—R44

R2

R1, R33

R20, R22

R14

R11

1K2

3K3

4K7

5K6

18K

22K

100K

220K

Rezistory TR 214:

| | |
|------|-----|
| R101 | 220 |
|------|-----|

Odporové trimry TP 041

R13, R16,

R5

R21, R23

R32

1K0

33K

47K

Konektory

| | |
|----|---------------|
| K2 | FRB TY5173211 |
|----|---------------|

Trafo

| | |
|-------|-------------------------|
| TR101 | 220 V/2×8 až 9 V, 0,5 A |
|-------|-------------------------|

Ostatní materiál

| | |
|-------|----------------------------------|
| L1 | cívka dle textu (6 PK 855 79,80) |
| S101 | síťový vypínač |
| Po101 | 50mA/250V |
| 1 ks | deska plošných spojů |
| 1 ks | chladič pro MA7805 |
| 1 ks | síťová šňůra |
| 1 ks | propojovací šňůra pro video |
| 1 ks | propojovací šňůra pro interface |
| 1 ks | pojistkové pouzdro |

Literatura:

- [1] Reček, J.: Teletext v Československu. Rozhlasová a televizní technika 3/1987, str. 79—89, VÚRT Praha.
- [2] Doušek, J.: Televizní informační služba. Elektronika 5/1987, str. 19—21.
- [3] Vlček, J.: Televizor Color 4428 pre rok 1990. Elektronika 2/1988, str. 16—17.
- [4] Břežanský, P., Mydlík, M.: Dekodér pre systém Teletext. Sdělovací technika 9/1985, str. 327—328.
- [5] Darrington, P., Daniels, J., F.: Wireless World Teletext decoder. Wireless World, November 1975, str. 498—504, December 1975, str. 563—566, January 1976, str. 37—42, February 1976, str. 47—51, March 1976, str. 75—69, April 1976, str. 64—68, May 1976, str. 64—68, June 1976, str. 53—55.
- [6] Russell, R., T.: Teletext decoder modification. Wireless World, December 1977, str. 36—41.
- [7] Hinton, J., H.: Character rounding for the Wireless World teletext decoder. Wireless World, November 1978, str. 49—53.
- [8] Mack, Z.: Příjem teletextových informací. Amatérské rádio řada A, 3/1988, str. 92—94, 4/1988, str. 134—136, 5/1988, str. 173—176.
- [9] Kyrš, F.: Deglitcher — obvod pro odrušení datových signálů. Amatérské rádio řada A, 11/1985, str. 421—422, 12/1985, str. 459—460.
- [10] TELETEX — návrh Československé státní normy; zpracoval ing. Jiří Reček, Čs. televize Praha, odbor technického rozvoje, březen 1988.

TELETEXT

Ing. Lumír Přibyl, Pavel Brychta,
V. Noska 67, 644 11 Zbýšov

Popis systému WST úrovně 1.5

1. Základní technické informace

1.1 TELETEXT je systém přenosu informací, který umožňuje přenášet stránky tvořené textem a grafickými symboly. Teletextový signál se přenáší v dosud nevyužitých řádcích puls-nímkového intervalu. Pokud není teletext doplněkem televizního obrazového signálu, je možný přenos signálu teletextu v libovolných tv řádcích.

1.2 Dekódované stránky se zobrazí buď samo-statně (tj. místo přijímaného televizního obra-zu), nebo mohou být prolínuty (resp. vloženy) přímo do televizního obrazu. Do televizního obrazu mohou být také vloženy titulky nebo zpravodajské vstupy. Pozn.: Při prolnutí jsou tv obraz a příslušná stránka zobrazeny současně přes sebe, při vložení se vytvoří v obrazu okno, ve kterém je zobrazena příslušná část stránky.

1.3 Každý datový řádek teletextu přenáší binární signál rychlosti 6,9375 Mbit/s. Tento systém přenosu lze využít v televizních systémech typu 625/50 s šířkou pásma pro přenos videosig-nálu min. 5,0 MHz.

1.4 Každý datový řádek přenáší informaci pro synchronizaci dat, informaci pro určení adresy a kódy pro řádek, který má 40 znaků.

1.5 Na stránce lze zobrazit až 25 řádků po 40 znacích, včetně zvláštního úvodního řádku, který se nazývá záhlaví, a posledního 25. řádku, který se zobrazí pouze za urči-tých podmínek (viz odst. 3.4.4.2).

1.6 Záhlaví obsahuje přidavné adresovací a řídící údaje, které jsou přenášeny místo kódů prv-ních osmi znaků. Tyto údaje umožňují identifi-kovat stránku a dále řídit způsob jejího zobraze-ní. Posledních osm znaků ze zbývajících 32 je vyhrazeno pro zobrazení reálného času.

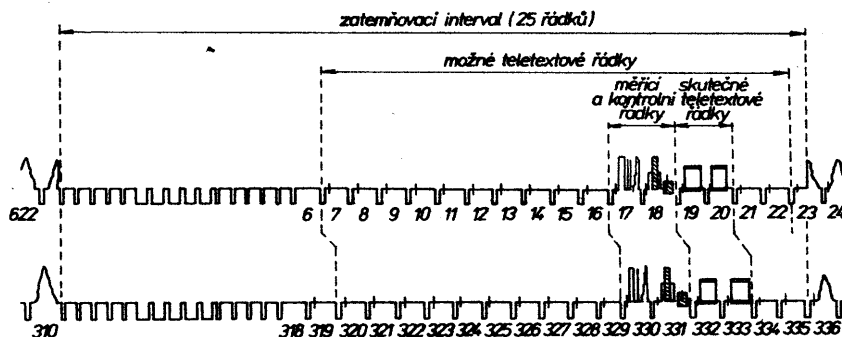
1.7 Přenos adresových a řídících bajtů využívá ochrany pomocí osmibitového Hammingova kódu. Tak lze v jednom bajtu přenést čtyři informační bity, přičemž jednoduché chyby jsou u přijímači opraveny. (Jednoduchá chyba znamená, že v jednom bajtu je z osmi bitů jeden chybný). Pro přenos kódů znaků je použito ochrany lichou paritou (sedm bitů je informačních, jeden bit je paritní), která umožňuje identifikovat jednoduché chyby.

1.8 Řádky, které obsahují pouze mezery, nemusí být přenášeny.

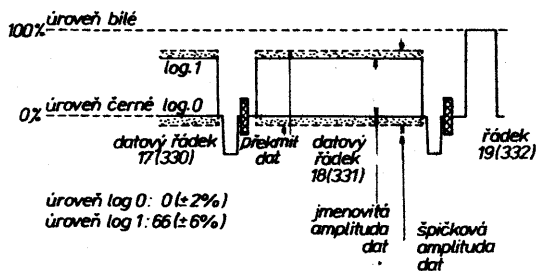
1.9 Lze přenášet až osm souborů stránek — magazínů, každý může obsahovat až 100 stránek.

1.10 Lze přenášet až 3200 variant každé stránky. Varianty jsou rozlišeny čtyřmístným sub-kódem v záhlaví stránky (viz odst. 3.1.2 a 3.1.3).

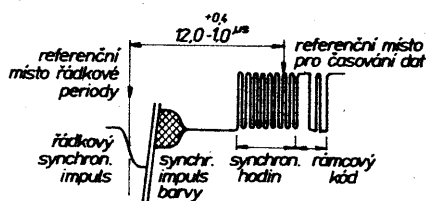
1.11 Kromě alfanumerických a grafických znaků se přenáší ještě řídící znaky, které slouží mimo jiné pro následující funkce:



Obr. 1. Umístění teletextových datových řádků v televizním signálu



Obr. 2. Úroveň signálu (čísla teletextových řádků nesouhlasí se skutečným umístěním v signálu ČST)



Obr. 3. Synchronizace dat (místo „referenční místo řádkové periody“ patří „referenční místo tv řádků“)

- a) volba jedné z osmi barev znaků,
- b) volba jedné z osmi barev pozadí,
- c) zobrazení znaků s dvojnásobnou výškou,
- d) blikání určitých znaků,
- e) skrytí určitých znaků, které mohou být uživatelem zviditelněny.

2. Teletextové datové řádky

2.1 Televizní obrazový signál obsahuje v oblasti puls-nímkového zatemňovacího intervalu (viz obr. 1) řádky číslo 7 až 22 (resp. 320 až 335). Tyto řádky jsou využívány pro různé účely. Některé z těchto řádků byly dosud volné a lze je využít pro přenos teletextových datových řádků. V případě ČST se jedná o řádky číslo 19, 20 (332,333). Obecně je možné využít libovolných volných řádků.

2.2 Obrazový řádek je identifikován jako teletextový datový řádek přítomností signálu pro synchronizaci hodin (viz odst. 2.9), po kterém následuje tzv. rámcový kód (viz odst. 2.10).

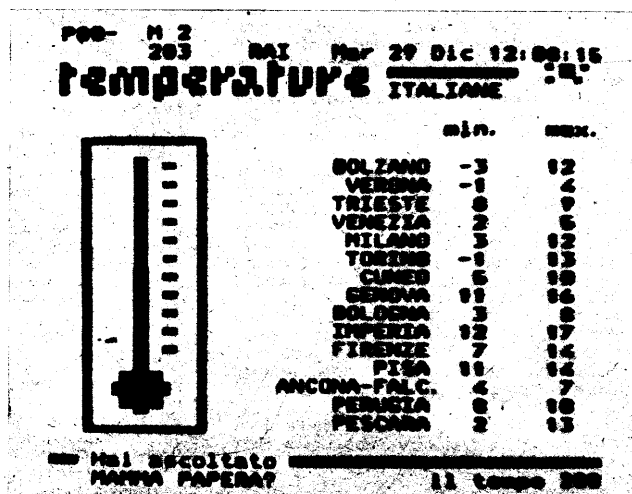
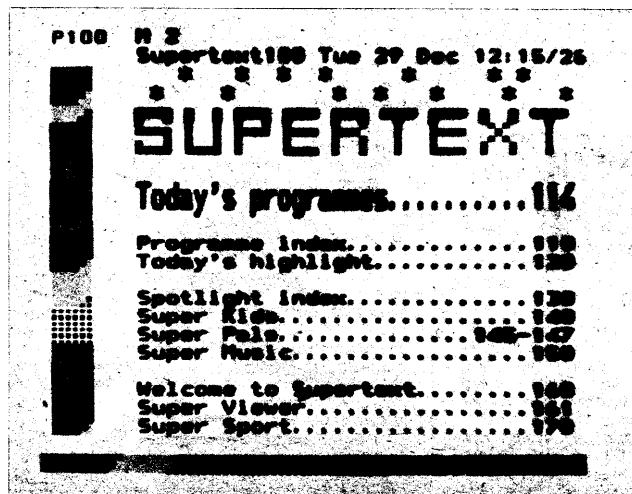
2.3 Teletextový řádek přenáší binární symboly (bity) ve formě signálu o dvou úrovních typu NRZ (Non-Return-to-Zero), vhodné tvarované pomocí filtru.

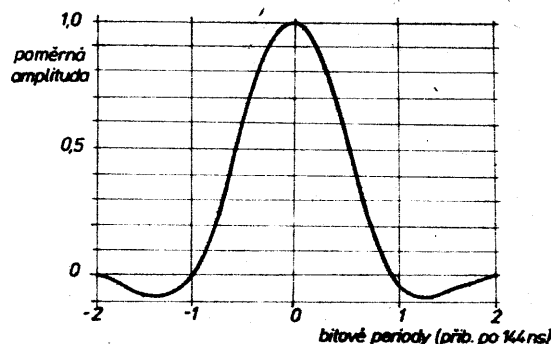
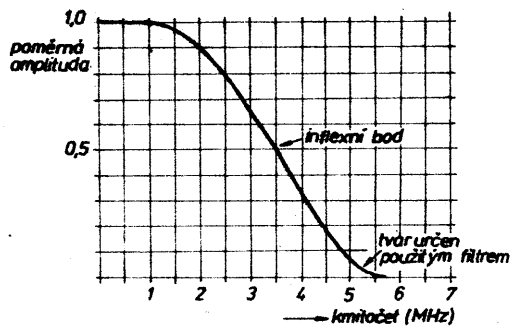
2.4 Úroveň signálu je taková, že logické nuly odpovídá úroveň černé $0 \pm 2\%$ a logické jedničky odpovídá $66 (\pm 6)\%$ z rozdílu mezi úrovní bílé a černé (viz obr. 2). Rozhodovací úroveň leží uprostřed mezi úrovní log. nuly a log. jedničky. Tato rozhodovací úroveň se může měnit řádek po řádku.

2.5 Bitová rychlost přenosu je $444 \times f_H$, kde f_H je řádková kmitočet. Bitová rychlost přenosu je tedy 6,9375 Mbit/s ($\pm 25.10^{-4}$).

2.6 Referenčním místem pro časování dat je předposlední logická jednička ve sledu impul-sů pro synchronizaci hodin (viz obr. 3). Referenční místo televizního řádku je v polovi-né amplitudě náběžné hrany řádkového syn-chronizačního impulsu. Referenční místo pro časování dat v signálu na výstupu vyslače distribuční sítě je zpožděno oproti refe-renčnímu místu televizního řádku o $12,0 + 0,4 - 1,0 \mu s$.

2.7 Spektrum signálu dat je asymetrické s inflexním bodem u kmitočtu, který odpovídá poloviční bitové rychlosti. Nad kmitočtem 5,0 MHz má spektrum prakticky nulovou hod-notu, viz obr. 4 a 5.





2.8 Datový rádek teletextu obsahuje 360 bitů, které tvoří 45 bajtů. První tři bajty mají sudou paritu a slouží pro synchronizaci přijímače. Zbývajících 42 bajtů má lichou paritu a přenášejí adresu, řídicí kódy a kódy znaků pro jeden rádek. Adresa a řídicí kódy jsou zabezpečeny Hammingovým kódem.

2.9 První dva bajty každého datového řádku jsou tvořeny posloupností jedniček a nul (101010...), indikují přítomnost datového řádku a určují časování bitů v datovém řádku. V některých případech mohou chybět v prvním bajtu první dvě log. jedničky.

2.10 Třetí bajt každého datového řádku obsahuje tzv. rámcový kód (11100100). Tento kód slouží k synchronizaci jednotlivých bajtů v příslušném datovém řádku při dekódování. Lze ho identifikovat i v případě, že jeden jeho bit byl chybně přijat. Obr. 6 ukazuje průchod dat posuvným registrem a jejich srovnání se vzorem rámcového kódu. Je zřejmé, že pro spolehlivé určení přítomnosti rámcového kódu stačí, aby souhlasilo nejméně 7 bitů.

2.11 Čtvrtý a pátý bajt každého datového řádku, dalších osm bajtů v záhlaví stránky (řádek 0) a všechny bajty přenášené v paketech 26 a 27, jsou chráněny pomocí Hammingova kódu.

2.12 Zbývající bajty v každém datovém řádku přenášejí kódy znaků. Bajt pro přenos znaku obsahuje 7 bitů určujících znak (viz obr. 15 Tabulka G0) a osmý bit pro zabezpečení lichou paritou. Nejprve se přenáší bit s nejnižší vahou (LSB).

3. Uspořádání stránek a řádků

X - hodnota těchto bitů závisí na čísle
magazínu a čísle řádku

Obr. 6. Průchod dat posuvným registrem

3.2 Vysílací pořadí

3.2.1 Vysílání vybrané stránky začíná přenosem záhlaví dané stránky (patří k dané stránce) a končí přenosem záhlaví následující stránky. Všechny datové řádky přenesené mezi těmito záhlavími a mající stejné číslo magazínu jako záhlaví vybrané stránky patří k této vybrané stránce. Stránky mohou být vysílány v libovolném pořadí. Mohou být přeneseny i nekompletní stránky. Mohou být proloženy datové řádky s jiným číslem magazínu.

3.2.2 Datové řádky patřící k dané stránce mohou být vysílány v libovolném pořadí. Aby byla zabezpečena co nejúčinnější funkce dekodérů, doporučuje se hned po záhlaví přenášet pakety 27 a pak pakety 26. Datové řádky včetně záhlaví se mohou opakovat, přičemž za platný se považuje poslední bezchybný výskyt příslušného datového řádku. Datové řádky, které obsahují nuluze mezery, nemusí být ategněny

3.2.3 Datové řádky jsou vysílány tak, aby byla vytvořena časová prodleva 20 ms (pro vymazání stránkové paměti některých dekodérů) mezi vysláním záhlaví a vysláním následujících datových řádků.

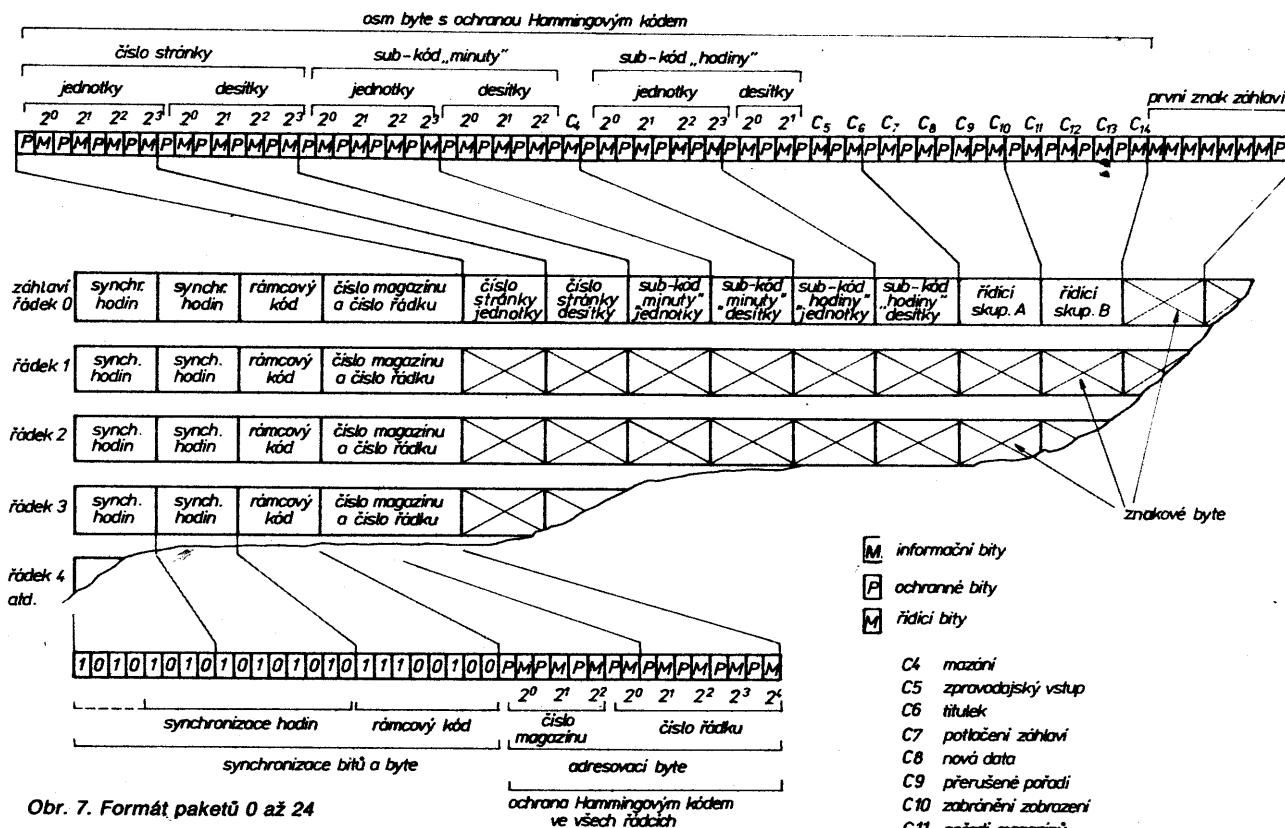
3.3 Záhľad

3.3.1 Číslo stránky, sub-kód stránky a informace nutné pro zobrazení stránky jsou přenášeny v tzv. záhlaví (řádek 0).

3.3.2 Synchronizace bitů, bajtů a adresování se provádí v 1. až 5. bajtu viz odst. 2.9, 2.10 a 3.1

3.3.3 Prvních osm bajtů datového řádku přenáší zhlávi (tj. 6. až 13. bajt) je chráněno Hammingovým kódem viz obr. 7. Tyto bajty obsahují informační bity vyznačující se k příslušné stránce včetně dvoucíselného čísla stránky a čtyřcíslného sub-kódu. Dále zhlávi obsahuje 32 kódů, ve kterých se přenáší zhlávi stránky určené pro zobrazení, tj. číslo magazínu a stránky, datum, název informační služby atd. Posledních osm znaků je vyhrazeno pro zobrazení reálného času. Příklady uspořádání zhlávi jsou na obr. 8.

3.3.4 V 6. a 7. bajtu se přenáší číslo stránky. V 8., 9., 10. a 11. bajtu se přenáší sub-kód stránky. Přitom 8. bit devátého bajtu je považován za řídicí bit C4, a 6. bit popř. 8. bit jedenáctého bajtu jsou považovány za řídicí bity C5 a C6. Řídicí bity C6 až C14 jsou přenášeny v 12. a 13. bajtu. Uspořádání 6. až 13. bajtu v záhlaví je na obr. 7.



3.3.5 Řídicí bity se považují za aktivní, jestliže mají hodnotu log. 1.

3.3.5.1 Bit C4 je aktivní v případě, že informace přenášené na dané stránce jsou zcela rozdílné od informací přenášených předchozí stránkou se stejným číslem magazínu a stránky. Při aktivaci tohoto bitu následuje interval 20 ms (viz 3.2.3), než se začnou přenášet data nové stránky.

3.3.5.2 Bit C5 je aktivní v případě, že daná stránka přenáší tzv. zpravodajský vstup („newsflash“), který se vloží do televizního obrazu. Veškerý text se zobrazí v tzv. vloženém módu viz odst. 4.2.10.

3.3.5.3 Bit C6 je aktivní v případě, že daná stránka přenáší titulky. Veškerý text se opět zobrazí v tzv. vloženém módu viz odst. 4.2.10.

3.3.5.4 Bit C7 je aktivní v případě, že je vhodnější potlačit zobrazení záhlaví dané stránky.

3.3.5.5 Bit C8 je aktivní v případě, když celá stránka nebo její část obsahuje novější informace než předchozí stránka se stejným číslem magazínu a stejným číslem strany. Stránka přenášená s tímto aktivovaným bitem nemusí být kompletní, může obsahovat pouze ty řádky, které mají být změněny.

3.3.5.6 Bit C9 je aktivní v případě, že daná stránka není v číselném pořadí ostatních přenášovaných stránek.

3.3.5.7 Bit C10 je aktivní v případě, že je třeba zabránit zobrazení dané stránky.

3.3.5.8 Bit C11 je aktivní v případě, že je vhodné v době čekání na navolenou stránku zobrazovat záhlaví všech magazínů a ne jenom záhlaví zvoleného magazínu.

3.3.5.9 Bity C12, 13, 14 volí základní soubor znaků viz obr. 20. Tabulka jazykových variant.

3.4 Paket 27

3.4.1 V paketu (řádku) 27 je možné přenášet adresy tzv. sdružených stránek, které jsou určeny k automatickému ukládání do paměti dekodéru.

3.4.2 Synchronizace bitů, bajtů a adresování se provádí v 1. až 5. bajtu viz odst. 2.9, 2.10 a 3.1.

3.4.3 V 6. bajtu je uložen určovací kód, který je chráněn Hammingovým kódem. Určovací kód pro aktivaci funkce sdružených stránek je 0000.

3.4.4. Adresování sdružených stránek. Bajty 7 až 42 jsou uspořádány do šesti skupin po šesti bajtech. Každá skupina šesti bajtů definuje adresu sdružené stránky. Tyto skupiny jsou číslovány 0 až 5 podle pořadí přenosu. Když se nastaví osmý bit (MSB) 43. bajtu na hodnotu 1 (viz odst. 3.4.4.2), jsou takto definované sdružené stránky označeny určitou barvou podle následující tabulky:

informační bity

P ochranné bity

M řidici bity

C4 *mazāni*

C5 zprůvodačský vstup

C6 titlelek

C7 potlače

CA novi data

C9 ព័ត៌មាន ផ្សេងៗ

C10 zadržanosti zadržanosti

C11 zažiti znanosti:

U17 potrdi magazinu

C12

C13 } jazyková varianta

skupina 0 „červená“

skupina 1 „zelená“

skupina 2 „žlutá“
skupina 2 „modrá“

skupina 3 „modrozelená“
skupina 4 nevyužitá

skupina 4 nevyuzita
skupina 5 seznam

stránek

3.4.4.1 Skupina dat pro definici sdružené stránky
je tvořena šesti bajty, které obsahují:

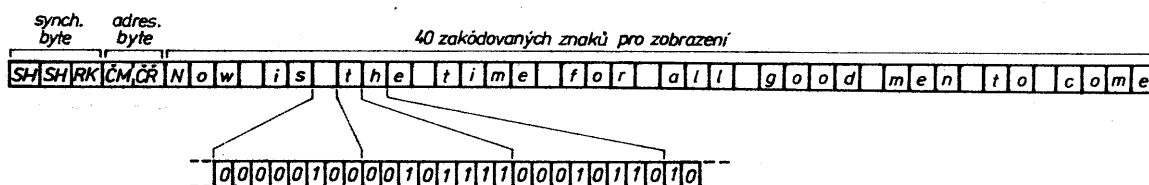
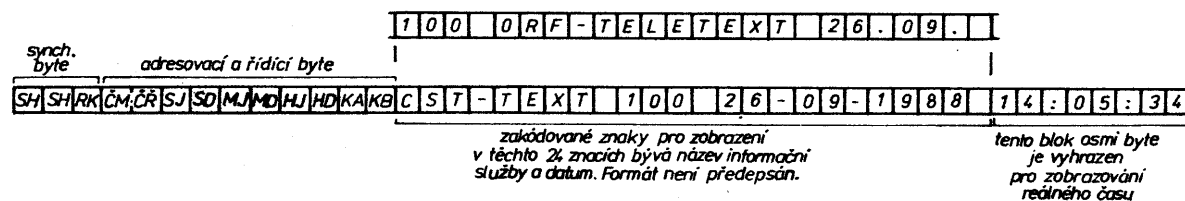
číslo vzťažného magazínu 3 bity

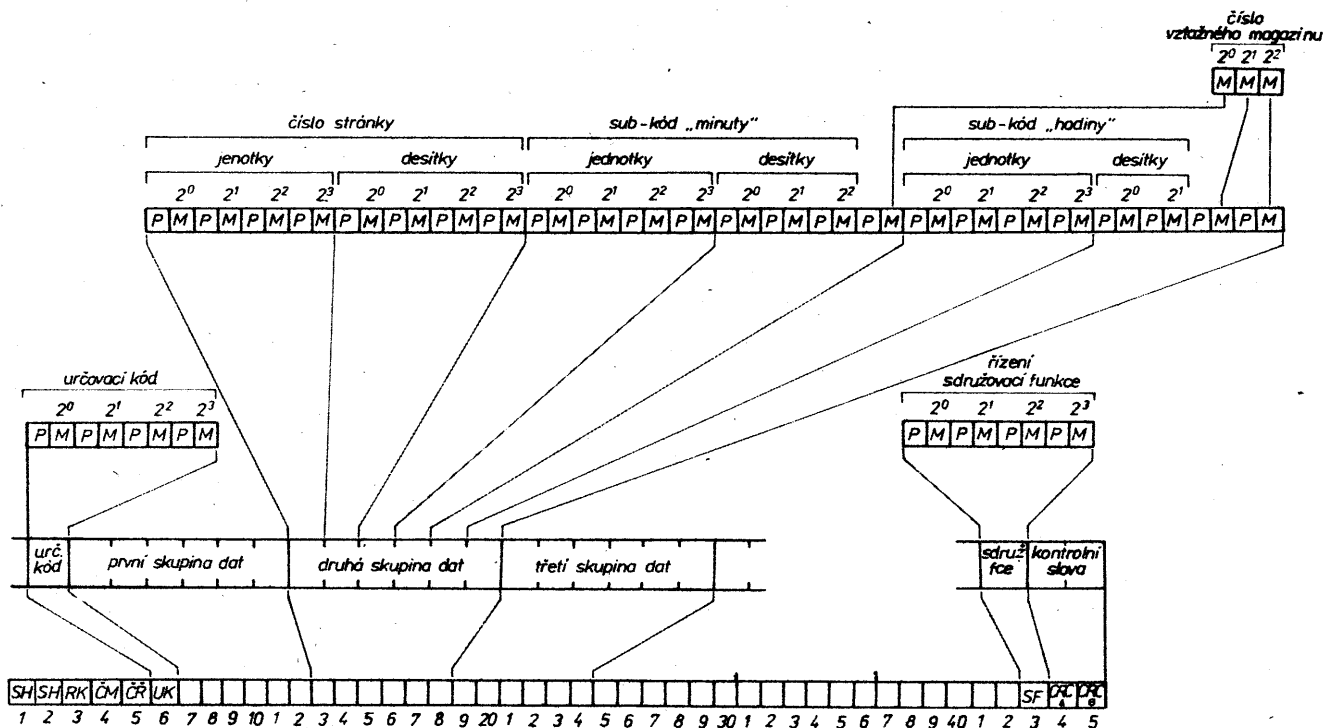
číslo stránky 8 bitů

| | |
|-------------------------------|---------|
| sub-kód stránky | 13 bitů |
| ochranné bity Hammingova kódu | 24 bitů |

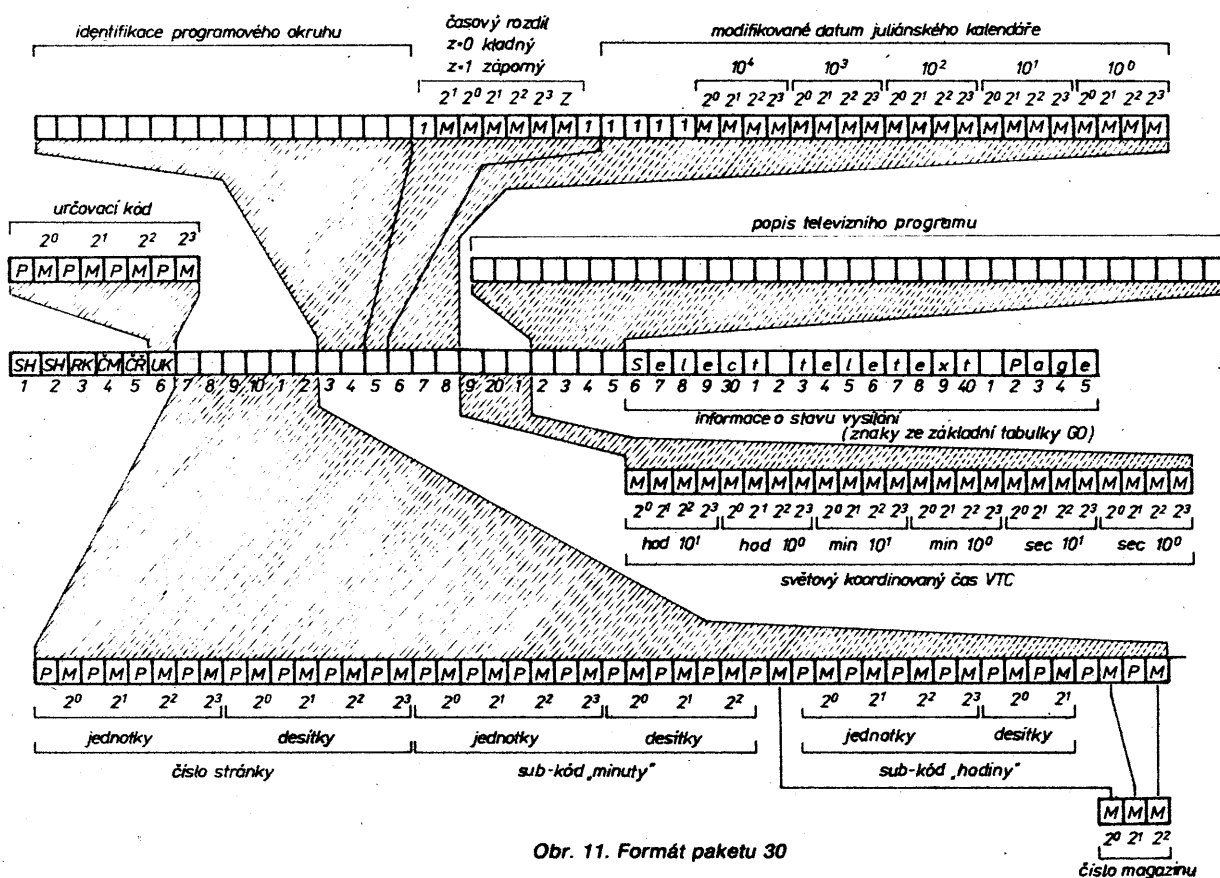
Pořadí bitů je na obr. 10. Nemá-li být specifikováno, platí následující pravidla:

Přádní bitů je na **adr. 10**. Nemá-li být specifikováno sub-kódu stránky, bude přenašena adresa sub-kódu 37F7 (hex). Přenáší-li se číslo stránky FF (hex), a sub-kód stránky 37F7, není specifikována žádná stránka. Bity, přenašené jako relativní číslo magazínu, se používají pro definici čísla magazínu sdružené stránky vzhledem k číslu magazínu v 4. bajtu paketu 27. Nastavení kteréhokoli z těchto bitů na „1“ způsobí doplnění příslušného bitu v číslu magazínu. V praxi to znamená provést mezi odpovídajícími bity čísla magazínu paketu 27 a čísla vztáženého magazínu log. operaci exclusive or. Výsledek je pak skutečné číslo magazínu sdružené stránky.





Obr. 10. Formát paketu 27



Obr. 11. Formát paketu 30

3.4.4.2 V bajtu 43 paketu 27 se přenáší informace pro řízení sdružovací funkce. Tento bajt má čtyři bity dat a čtyři bity ochrany Hammingovým kódem. Když je osmý bit (MSB) nastaven na hodnotu 1, znamená to, že sdružené stránky mají navazovat na symboly přegášené paktem 24. Pak je červené tlačítko vždy funkční a pokud jsou bity 2, 4 a 6 nastaveny na hodnotu 1, jsou funkční i tlačítka zelená, žlutá. resp.

modrozelené. Tlačítko pro navolení seznamu (indexu) je vždy funkční. (Tato tlačítka jsou součástí ovladačů některých dekoderů). Když je bit 8 nastaven na 0, pak neexistují sdružené stránky odpovídající barevným symbolům, a zamezí se zobrazení obsahu eventuálně přítomného paketu 24.

3.4.5 Bajty 44 a 45 obsahují kontrolní slovo pro cyklickou kontrolu nadbytečnosti (CRC) dat v paketech 0 až 25. Způsob generování tohoto slova je uveden v kap. 6.

3.5 Paket 30

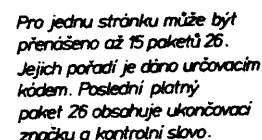
3.5.1. V paketu 30 se přenáší data, týkající se televizní organizace. Přenáší se přibližně jed-

nou za sekundu s číslem magazínu 8. Formát paketu 30 je na obr. 11.

3.5.2 Synchronizace bitů, bajtů a adresování se provádí v 1. až 5. bajtu viz odst. 2.9, 2.10 a 3.1.

3.5.3 V 6. bajtu je uložen určovací kód, který je chráněn Hammingovým kódem. Pokud je druhý bit nastaven na 0, znamená to multiplexovanou funkci teletextu v zatemňovacím intervalu televizního obrazového signálu (viz odst. 1.1). Pokud jsou bity 4, 6 a 8 nastaveny na hodnotu 0, jsou aktivní funkce popsané v následujících odstavcích 3.5.3.1 až 3.5.3.6. Na jiné kódy systém nereaguje.

3.5.3.1 Bajty 7 až 12 obsahují adresu první vybrané stránky, která bude uložena do paměti



TEKNOLOGIJA

Seznam datových bitů definuje znak z doplňkové tabulky (obr. 16) ze sloupce 2, 3, 5, 6, 7.

3.6.7 Jelikož může být použito většího počtu paketů 26 na jedné stránce, je třeba označit poslední paket. Toho se docílí nastavením všech bitů pro adresu řádku a popis módu na hodnotu 1, a to ve skupině dat přenášené v 40., 41. a 42. bajtu konečného paketu 26. Takováto skupina vytváří ukončovací značku. Na data v 42. bajtu systém nereaguje. Nevyužitá skupiny dat mezi poslední aktivní skupinou a ukončovací značkou se vyplní opakujícími se daty ukončovací značky.

3.6.8 Dva bajty, které následují za skupinou obsahující ukončovací značku, je možné využít pro cyklickou kontrolu nadbytečnosti (CRC) dat v paketech 26. Kontrolní slovo se vytváří stejným způsobem jako kontrolní slovo CRC v paketech 27 (viz kap. 6), ovšem za použití dat z paketů 26. V případě chybějících dat se předpokládá, že to jsou znaky pro mezeru.

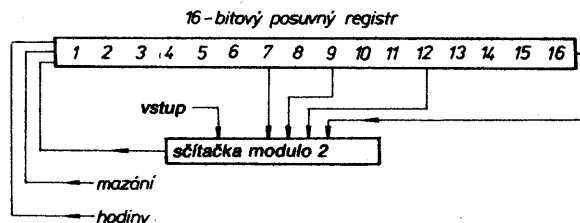
4. Zobrazení stránky

- 4.1 Stránka zobrazená přijímačem obsahuje až 25 řádků.
- 4.1.1 Zobrazené řádky odpovídají paketům dat 0 až 24. Řádek 24 se zobrazí pouze tehdy, je-li přenášén také paket 27, a 8. bit 43. bajtu je nastaven na 1. Řádek 24 může obsahovat barevné symboly, které usnadňují přístup k sdruženým stránkám. Přitom může být využíváno příslušných barevných tlačítek na jednotce dálkového ovládání.
- 4.1.2 Na řádku 1 až 24 je 40 míst pro znaky, na řádku 0 je 32 míst pro znaky. Časový přenos probíhá zleva doprava.
- 4.2 Znaky zobrazené na řádku 0 mohou být všechny znaky z tabulky G0. Znaky zobrazené na řádcích 1 až 24 mohou být všechny znaky z tabulky G0 a G2. Znaky mohou být zobrazeny v různých zobrazovacích módech. Volba jednotlivých módů je určena implicitně na začátku jednotlivých řádků, explicitně se provádí v průběhu zobrazování řádku pomocí řídících znaků (obr. 17). Některé řídící znaky mají bezprostřední účinek, jiné působí až na následující místo pro znak. Řídící znaky jsou vesměs zobrazeny jako mezery v barvě pozadí.
- 4.2.1 Barva znaků (alfanumerických i grafických) může být bílá, žlutá, modrozelená, zelená, purpurová, červená, modrá a černá. Příslušné dvojice řídících znaků zároveň přepínají zobrazení v alfanumerickém a grafickém módu.
- 4.2.3 Nové pozadí. Předcházející barva znaků se stává barvou nového pozadí.
- 4.2.4 Černé pozadí. Nastaví se černá barva pozadí.
- 4.2.5 Spojitá grafika. Grafické prvky přiléhají jeden k druhému bez mezery.
- 4.2.6 Oddělená grafika. Grafické prvky jsou odděleny mezerami (proučky) v barvě pozadí (obr. 13).
- 4.2.7 Seřazená grafika. Seřazená grafika umožňuje mezeru obsahující řídící znak překrýt grafickým znakem. Tento znak je definován pouze v grafickém módu. Je to poslední znak, který předchází před danou mezerou a jehož 6. bit má hodnotu 1. Přitom nesmí dojít ke změně alfanumerika/grafika nebo jednoduchá/dvojité výška. Znak seřazené grafiky se zobrazí jako spojitý nebo oddělený ve shodě s nastaveným módem.
- 4.2.8 Skrytí a zviditelnění znaků. Znaky, které jsou vymezeny pomocí těchto řídících znaků, se zobrazují jako mezery do doby, než jsou zviditelněny činností dekódovacího zařízení nebo zásahem uživatele.
- 9 Blikání a stálé zobrazení (konec blikání). Znaky, které jsou vymezeny pomocí těchto řídících znaků, se zobrazují střídavě jako znaky a jako mezery v barvě pozadí. Kmitočet blikání je určen dekódovacím zařízením.
- 4.2.10 Začátek vkládání a konec vkládání. Definuje se část obrazu určená pro vložení do normálního tv obrazu (titulky, zpravodajský vstup). Tato funkce je chráněna zdvojeným přenosem řídících znaků označujících začátek a konec vkládání.
- 4.2.11 Dvojité výška. Znaky jsou prodlouženy vertikálně do prostoru následujícího řádku. Zobrazení v druhém řádku má stejnou barvu znaků i pozadí jako první řádek. Případné znaky přenášené v druhém řádku jsou ignorovány.

5. Zabezpečení dat Hammingovým kódem

Přenos některých dat je proti chybám zabezpečen Hammingovým kódem. Používá se dvou modifikací tohoto zabezpečení:

Obr. 14. Vytvoření kontrolního slova



5.1 Jeden bajt obsahující čtyři informační bity a čtyři zabezpečovací (ochranné) bity (varianta A). Zabezpečovací jsou bity 1, 3, 5, 7, informační jsou bity 2, 4, 6, 8. Zabezpečovací bity se pro danou kombinaci informačních bitů doplňují podle tabulky Hammingova kódu obr. 21a. Při dekódování se nejprve provedou následující operace:

A = b8 xor b6 xor b2 xor b1
B = b8 xor b4 xor b3 xor b2
C = b6 xor b5 xor b4 xor b2
D = b8 xor b7 xor b6 xor b5 xor b4 xor b3 xor b2 xor b1

xor označuje logickou operaci exclusive or

Graficky jsou testované bity vyznačeny v tabulce obr. 21b. Výsledky těchto operací se pokládají za správné tehdy, jestliže jsou rovny log. 1. Prakticky to znamená, že vybraná skupina bitů musí mít lichou paritu. Vyhodnocení chyb se provádí podle tabulky obr. 21c.

5.2 Skupiny tří bajtů obsahující 18 informačních bitů a 6 zabezpečovacích (ochranných) bitů (varianta B). Zabezpečovací jsou bity 1, 2, 4, 8, 16 a 24, informační jsou bity 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17 až 23. Zabezpečovací bity se pro danou kombinaci informačních bitů vypočtou podle následujících vztahů:

b1 = not (b23 xor b21 xor b19 xor b17 xor b15 xor b13 xor b11 xor b9 xor b7 xor b5 xor b3)
b2 = not (b23 xor b22 xor b19 xor b18 xor b15 xor b14 xor b11 xor b10 xor b7 xor b6 xor b3)
b4 = not (b23 xor b22 xor b21 xor b20 xor b15 xor b14 xor b13 xor b12 xor b7 xor b6 xor b5)
b8 = not (b15 xor b14 xor b13 xor b12 xor b11 xor b10 xor b9)
b16 = not (b23 xor b22 xor b21 xor b20 xor b19 xor b18 xor b17)
b24 = not (b23 xor b22 xor b21 xor b20 xor b19 xor b18 xor b17 xor b16 xor b15 xor b14 xor b13 xor b12 xor b11 xor b10 xor b9 xor b8 xor b7 xor b6 xor b5 xor b4 xor b3 xor b2 xor b1)

Při dekódování se nejprve provedou následující operace:

A = b23 xor b21 xor b19 xor b17 xor b15 xor b13 xor b11 xor b9 xor b7 xor b5 xor b3 xor b1
B = b23 xor b22 xor b19 xor b18 xor b15 xor b14 xor b11 xor b10 xor b7 xor b6 xor b3 xor b2
C = b23 xor b22 xor b21 xor b20 xor b15 xor b14 xor b13 xor b12 xor b7 xor b6 xor b5 xor b4
D = b15 xor b14 xor b13 xor b12 xor b11 xor b10 xor b9 xor b8
E = b23 xor b22 xor b21 xor b20 xor b19 xor b18 xor b17 xor b16
F = b24 xor b23 xor b22 xor b21 xor b20 xor b19 xor b18 xor b17 xor b16 xor b15 xor b14 xor b13 xor b12 xor b11 xor b10 xor b9 xor b8 xor b7 xor b6 xor b5 xor b4 xor b3 xor b2 xor b1

xor označuje logickou operaci exclusive or

Graficky jsou testované bity vyznačeny v tabulce obr. 22a. Výsledky těchto operací se pokládají za správné tehdy, jestliže jsou rovny log. 1. To znamená, že vybraná skupina bitů musí mít lichou paritu. Vyhodnocení chyb se provádí podle tabulky obr. 22b.

6. Vytvoření kontrolního slova

Vytvoření kontrolního slova je znázorněno na obr. 14. Na vstup 16-ti bitového posuvného registru je přiváděn výsledek součtu modulo 2 externího vstupu a obsahu 7., 9., 12. a 16. stupně registru. Na začátku se registr vynuluje ve všech stupních. Během sekvence 8192 hodinových impulsů je vstupní signál tvořen prvními 24 znakovými bajty (24 x 8, tj. 192 bitů) ze záhlaví (řádku 0) a následujícími znakovými bajty z řádků 1 až 25 (40 x 8 x 25, tj. 8000 bitů), a to při normálním pořadí přenosu. Chybí-li některý paket, bere se jako by obsahoval vesměs znaky 'SP' (mezera) (20). V každém bajtu je pořadí bitů b8 až b1. Toto pořadí, které je opačné než při vysílání ostatních bajtů textu, slouží pro usnadnění činnosti dekódu, který využívá data uložená ve stránkové paměti.

| | 2 | 3 | 4 ⁽¹⁾ | 5 | 6 | 7 |
|----|----|---|------------------|---|----|-----|
| 0 | SP | ° | — | Ω | K | |
| 1 | ; | ± | ˘ | Æ | æ | |
| 2 | £ | ² | ˙ | ⊗ | Ⓢ | α |
| 3 | £ | ³ | ˘ | ⊙ | Ⓢ | δ |
| 4 | £ | ˣ | ˘ | ™ | ℥ | ℥ |
| 5 | ¥ | μ | — | ℥ | | ı |
| 6 | # | π | — | ℥ | ıı | ıı |
| 7 | § | — | — | ‰ | ℥ | ℥ |
| 8 | π | ÷ | — | α | ℥ | ℥ |
| 9 | ‘ | ’ | — | — | ø | ø |
| 10 | “ | ” | — | — | œ | œ |
| 11 | « | » | — | — | — | — |
| 12 | ← | ¼ | — | ½ | Ⓢ | Ⓢ |
| 13 | † | ½ | — | ¾ | ℥ | ℥ |
| 14 | → | ¾ | — | ¾ | Ⓢ | η |
| 15 | ı | ı | ı | ¾ | ı | (2) |

Obr. 16. Tabulka G2

(1) Sloupec 4 obsahuje diakritická znaménka, která se přiřazují znakům z tabulky G0 (obr. 15). (2) Znak 7/15 je interpretován stejně jako v tabulce G0. Kód 4/0 je „nulové znaménko“, což umožňuje zobrazení libovolného symbolu z tabulky G0 bez diakritického znaménka s využitím paketu 26. Kódy 4/9 a 4/12 jsou rezervovány pro budoucí využití.

| | S0 | S1 | 2 | 3 | 4 | 5 | 6 | 7 |
|----|----|----|---|---|---|---|---|---|
| 0 | Ü | ó | □ | 0 | č | P | é | p |
| 1 | ° | Č | ! | 1 | A | Q | a | q |
| 2 | ä | Đ | “ | 2 | B | R | b | r |
| 3 | Á | đ | # | 3 | C | S | c | s |
| 4 | Š | L | ů | 4 | D | T | d | t |
| 5 | É | Í | % | 5 | E | U | e | u |
| 6 | Ê | Ĺ | & | 6 | F | V | f | v |
| 7 | Í | Ĺ | ' | 7 | G | W | g | w |
| 8 | ö | Ň | (| 8 | H | X | h | x |
| 9 | ü | ň |) | 9 | I | Y | i | y |
| 10 | B | Ř | * | : | J | Z | j | z |
| 11 | Ä | ř | + | ; | K | ř | k | á |
| 12 | Ö | Ř | . | < | L | ž | l | ě |
| 13 | Ü | ř | — | = | M | ý | m | ú |
| 14 | Ó | Ý | . | > | N | í | n | š |
| 15 | ó | Ž | / | ? | O | ř | o | ■ |

Znaky přenášené pomocí paketu 26

Národní tabulka čs. znaků

Obr. 18. Úplný soubor znaků pro češtinu a slovenštinu

Poznámka: Tento soubor znaků byl doplněn ještě o znaky Ō a Ŭ.

Na konci procesu plnění registru je obsah registru tvořen kontrolním slovem základní stránky.

Pořadí přenosu skupiny se dvěma bajty, která vznikne 16-ti bitovou cyklickou kontrolou nadbytečnosti dané stránky, je bit 9 až 16 (CRC A) a poté bit 1 až 8 (CRC B) včetně.

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | sloup. | 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
|---------|----|----|----|----|----|----|--------|--------------------------------------|-------------------------|-------|-------|-------|-------|-------|-------|
| bity | b7 | b6 | b5 | b4 | b3 | b2 | b1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | alfan. černá | grafika černá | | | | | | |
| 0 0 0 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | alfan. červená | grafika červená | | | | | | |
| 0 0 1 0 | 2 | 0 | 0 | 1 | 0 | 0 | 2 | alfan. zelená | grafika zelená | | | | | | |
| 0 0 1 1 | 3 | 0 | 0 | 1 | 1 | 0 | 3 | alfan. žlutá | grafika žlutá | | | | | | |
| 0 1 0 0 | 4 | 0 | 1 | 0 | 0 | 0 | 4 | alfan. modrá | grafika modrá | | | | | | |
| 0 1 0 1 | 5 | 0 | 1 | 0 | 1 | 0 | 5 | alfan. purpur. | grafika purpur. | | | | | | |
| 0 1 1 0 | 6 | 0 | 1 | 1 | 0 | 0 | 6 | alfan. modroz. | grafika modroz. | | | | | | |
| 0 1 1 1 | 7 | 0 | 1 | 1 | 1 | 0 | 7 | alfan. bílá | grafika bílá | | | | | | |
| 1 0 0 0 | 8 | 1 | 0 | 0 | 0 | 0 | 8 | blíkání | skrytí | | | | | | |
| 1 0 0 1 | 9 | 1 | 0 | 0 | 1 | 0 | 9 | stálé ^{*)} zobraz. | spojitá grafika | | | | | | |
| 1 0 1 0 | 10 | 1 | 0 | 1 | 0 | 0 | 10 | konec ^{*)} vkládání | odděl. grafika | | | | | | |
| 1 0 1 1 | 11 | 1 | 0 | 1 | 1 | 0 | 11 | začátek vkládání norm. ^{*)} | ESC ^{*)} černé | | | | | | |
| 1 1 0 0 | 12 | 1 | 1 | 0 | 0 | 0 | 12 | výška dvoj. | nové pozad. | | | | | | |
| 1 1 0 1 | 13 | 1 | 1 | 0 | 1 | 0 | 13 | výška sevř. | grafika uvolněn. | | | | | | |
| 1 1 1 0 | 14 | 1 | 1 | 1 | 0 | 0 | 14 | S0 ^{*)} | | | | | | | |
| 1 1 1 1 | 15 | 1 | 1 | 1 | 1 | 0 | 15 | S1 ^{*)} | | | | | | | |

vyhrazeno pro národní varianty znaků

*) v současné úrovni teletextu se nepoužívají

**) předpokládá se na začátku každého řádku

černá představuje barvu znaku, bílá představuje barvu pozadí
(znaky ve sloupci 4 a 5 mohou být zobrazeny i v grafickém módu — tzv. pronikání alfanumerických znaků)

Obr. 15. Tabulka G0

| MÓD ZOBRAZENÍ | OKAMŽITÉ PŮSOBNÍ | NÁSLEDNÉ PŮSOBNÍ | DOPLŇKOVÝ MÓD ZOBRAZENÍ | OKAMŽITÉ PŮSOBNÍ | NÁSLEDNÉ PŮSOBNÍ |
|-------------------------------|--------------------------------|---|-------------------------|------------------|------------------|
| Alfanumerika bílá | Zač. řádku | 0/7 | Grafika bílá | — | 1/7 |
| červená | — | 0/1 | červená | — | 1/1 |
| zelená | — | 0/2 | zelená | — | 1/2 |
| modrá | — | 0/4 | modrá | — | 1/4 |
| žlutá | — | 0/3 | žlutá | — | 1/3 |
| purpurová | — | 0/5 | purpurová | — | 1/5 |
| modrozelená | — | 0/6 | modrozelená | — | 1/6 |
| černá | — | 0/0 | černá | — | 1/0 |
| Spojitá (grafika) | Zač. řádku 1/9 * | 1/9 * | Oddělená (grafika) | 1/10 * | 1/10 * |
| Černé pozadí | Zač. řádku 1/12 | — | Nové pozadí | 1/13 ** | — |
| Zviditelnění znaků | Zač. řádku Zásah uživatele *** | 0/1 0/2 0/3 0/4 0/5 0/6 0/7 1/1 1/2 1/3 1/4 1/5 1/6 1/7 0/0 1/0 | Skrytí znaků | 1/8 | — |
| Stálé zobrazení konec blíkání | Zač. řádku 0/9 | — | Blíkání | — | 0/8 |
| Konec vkládání | Zač. řádku 0/10 **** | 0/10 **** | Začátek vkládání | 0/11 **** | 0/11 **** |
| Normální výška | Zač. řádku 0/12 | — | Dvojitá výška | — | 0/13 |
| Uvolněná grafika | Zač. řádku | 1/15 | Sevřená grafika | 1/14 | — |

* Tyto řídicí znaky mohou působit okamžitě nebo následně

** Aktuální barva znaků je vzata jako nová barva pozadí

*** Zviditelnění skrytých znaků na celé stránce může být provedeno zásahem uživatele

**** Tyto řídicí znaky jsou přenášeny zdvojené; první působí následně, druhý okamžitě

Obr. 17. Tabulka působení řídicích znaků

| Pozice v tabulce G0 | Angličtina | Němčina | Čeština/na/slovenština | Švédština |
|---------------------|------------|---------|------------------------|-----------|
| 2/3 | £ | # | # | # |
| 2/4 | \$ | \$ | û | o |
| 4/0 | @ | \$ | č | É |
| 5/11 | - | Ä | ř | Ä |
| 5/12 | 1/2 | Ö | ž | Ö |
| 5/13 | → | Ü | ý | Ä |
| 5/14 | † | Λ | í | Ü |
| 5/15 | # | — | ř | — |
| 6/0 | - | ° | é | é |
| 7/11 | 1/4 | ä | á | ä |
| 7/12 | ll | ö | ě | ö |
| 7/13 | 3/4 | ü | ú | ä |
| 7/14 | ÷ | ß | š | ü |

Obr. 19. Národní znaky tabulky G0

| C12 | C13 | C14 | Západní Evropa | Východní Evropa |
|-----|-----|-----|----------------------------|----------------------------|
| 0 | 0 | 0 | angličtina | poštiná |
| 0 | 0 | 1 | němčina | němčina |
| 0 | 1 | 0 | švédština, finština | srbochorvatština |
| 0 | 1 | 1 | italština | italština |
| 1 | 0 | 0 | francouzština | francouzština |
| 1 | 0 | 1 | portugalština, španělština | portugalština, španělština |
| 1 | 1 | 0 | čeština, slovenština | čeština, slovenština |
| 1 | 1 | 1 | maďarština | maďarština |

Obr. 20. Tabulka jazykových variant

INFORMAČNÍ BITY

| ZABEZPEČOVACÍ BITY | | | | | | | |
|--------------------|----|----|----|----|----|----|----|
| b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |

Obr. 21a. Tabulka Hammingova kódu pro jeden bajt

| | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 |
|---|----|----|----|----|----|----|----|----|
| A | 0 | x | 0 | x | x | x | 0 | 0 |
| B | 0 | x | x | x | 0 | 0 | 0 | x |
| C | x | x | 0 | 0 | 0 | x | 0 | x |
| D | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

o Testované bity

Obr. 21b. Testy na lichou paritu (varianta A)

| Výsledky testů parity | | Vyhodnocení | Činnost |
|-----------------------|------------|----------------------|---|
| A, B, C | D | | |
| Všechny správně | Správně | Žádná chyba | Použít informační bity |
| Všechny správně | Ne-správně | Chyba v b7 | Použít informační bity |
| Ne všechny správně | Správně | Několiknásobná chyba | Odmítnout informační bity |
| Ne všechny správně | Ne-správně | Jednoduchá chyba | Srovnáním s tabulkou obr. 21b identifikovat chybu. Jestliže je v informačním bitu, pak opravit. Použít informační bity. |

Obr. 21c. Tabulka vyhodnocení chyb (varianta A)

| | A | B | C | D | E | F |
|-----|---|---|---|---|---|---|
| b1 | 0 | X | X | X | X | 0 |
| b2 | X | 0 | X | X | X | 0 |
| b3 | 0 | 0 | X | X | X | 0 |
| b4 | X | X | 0 | X | X | 0 |
| b5 | 0 | X | 0 | X | X | 0 |
| b6 | X | 0 | 0 | X | X | 0 |
| b7 | 0 | 0 | 0 | X | X | 0 |
| b8 | X | X | X | 0 | X | 0 |
| b9 | 0 | X | X | 0 | X | 0 |
| b10 | X | 0 | X | 0 | X | 0 |
| b11 | 0 | 0 | X | 0 | X | 0 |
| b12 | X | X | 0 | 0 | X | 0 |
| b13 | 0 | X | 0 | 0 | X | 0 |
| b14 | X | 0 | 0 | 0 | X | 0 |
| b15 | 0 | 0 | 0 | 0 | X | 0 |
| b16 | X | X | X | X | 0 | 0 |
| b17 | 0 | X | X | X | 0 | 0 |
| b18 | X | 0 | X | X | 0 | 0 |
| b19 | 0 | 0 | X | X | 0 | 0 |
| b20 | X | X | 0 | X | 0 | 0 |
| b21 | 0 | X | 0 | X | 0 | 0 |
| b22 | X | 0 | 0 | X | 0 | 0 |
| b23 | 0 | 0 | 0 | X | 0 | 0 |
| b24 | X | X | X | X | X | 0 |

0 — Testované bity

Obr. 22a. Testy na lichou paritu (varianta B)

7. Závěr

Protože při označování řádků, bajtů a bitů je používáno číslování od 0 včetně, mohou vzniknout zdánlivé nesrovnalosti mezi tímto popisem a jinými články, které se zabývají touto problematikou. Dále je třeba vzít v úvahu, že celý systém je stále ještě ve vývoji a může se v některých detailech měnit. A konečně, omlouváme se za všechny nepřesnosti a nedostatky (např. neúplná tabulka G2, popis paketu 30 atd.), které jsou způsobeny tím, že se přes veškerou snahu nepodařilo zajistit potřebné zdroje.

Závěrem bychom chtěli poděkovat ing. Jiřímu Rečkové z Československé televize za velice účinnou pomoc při sestavování tohoto článku.

| Výsledky testů parity | | Vyhodnocení | Činnost |
|-----------------------|------------|----------------------|--|
| A, B, C, D, E | F | | |
| Všechny správně | Správně | Žádná chyba | Použít informační bity |
| Všechny správně | Ne-správně | Chyba v b24 | Použít informační bity |
| Ne všechny správně | Správně | Několiknásobná chyba | Odmítnout informační bity |
| Ne všechny správně | Ne-správně | Jednoduchá chyba | Srovnáním s tabulkou obr. 22a) identifikovat chybu. Jestliže je v informačním bitu, pak opravit. Použít informační bity. |

Obr. 22b. Tabulka vyhodnocení chyb (varianta B)

Literatura

- [1] Reček, J.: Teletext v Československu. VÚRT Praha — Rozhlasová a televizní technika 3/1987, str. 79—89.
- [2] Broadcast Teletext Specification, BBC — IBA — BREMA, September 1976.
- [3] TELETEXT — návrh Československé státní normy. Zpracoval ing. Jiří Reček, Čs. televize Praha, odbor technického rozvoje, březen 1988.
- [4] Darrington, P., Daniels, J., F.: Wireless World Teletext decoder. Wireless World, November 1975, str. 498—504, December 1975, str. 563—566, January 1976, str. 37—42, February 1976, str. 47—51, March 1976, str. 75—79, April 1976, str. 64—68, May 1976, str. 64—68, June 1976, str. 53—55.
- [5] Russell, R., T.: Teletext decoder modification. Wireless World, December 1977, str. 36—41.
- [6] Mack, Z.: Přijem teletextových informací. Amatérské rádio řada A, 3/1988, str. 92—94, 4/1988, str. 134—136, 5/1988, str. 173—176.

PROGRAM TELETEXT

ing. Lumír PŘIBYL, Pavel BRYCHTA, v. Noska 67, 664 11 Zbýšov

Program TELETEXT má za úkol zabezpečit takové zpracování dat přicházejících z adaptéru, aby výsledná soustava, využívající adaptéru a mikropočítače pro příjem teletextu, plnila všechny funkce běžné u standardních dekodérů. Základní technická data programu ve spojení s adaptérem jsou uvedena v následujících bodech:

1. Program umožňuje příjem teletextu systému WST úrovně 1.5. Je využíváno paketů 26 a 27. Paket 30 a kontrolní slova CRC nejsou využity.
2. Zobrazovaná stránka obsahuje 24 řádků po 40. znacích. Je možné zobrazit i doplňkový 25. řádek. Jsou zobrazovány všechny znaky národních abeced češtiny/slovenštiny, němčiny a švédštiny.

3. Program umožňuje využít všech zvláštních funkcí běžných standardních dekodérů (volba stránky, přímá volba podstránky, podržení rotující podstránky, zobrazení v dvojité výšce, skryté zobrazení).
4. Program neumožňuje vkládání titulků do televizního obrazu.
5. Program umožňuje tisk zvolených stránek.

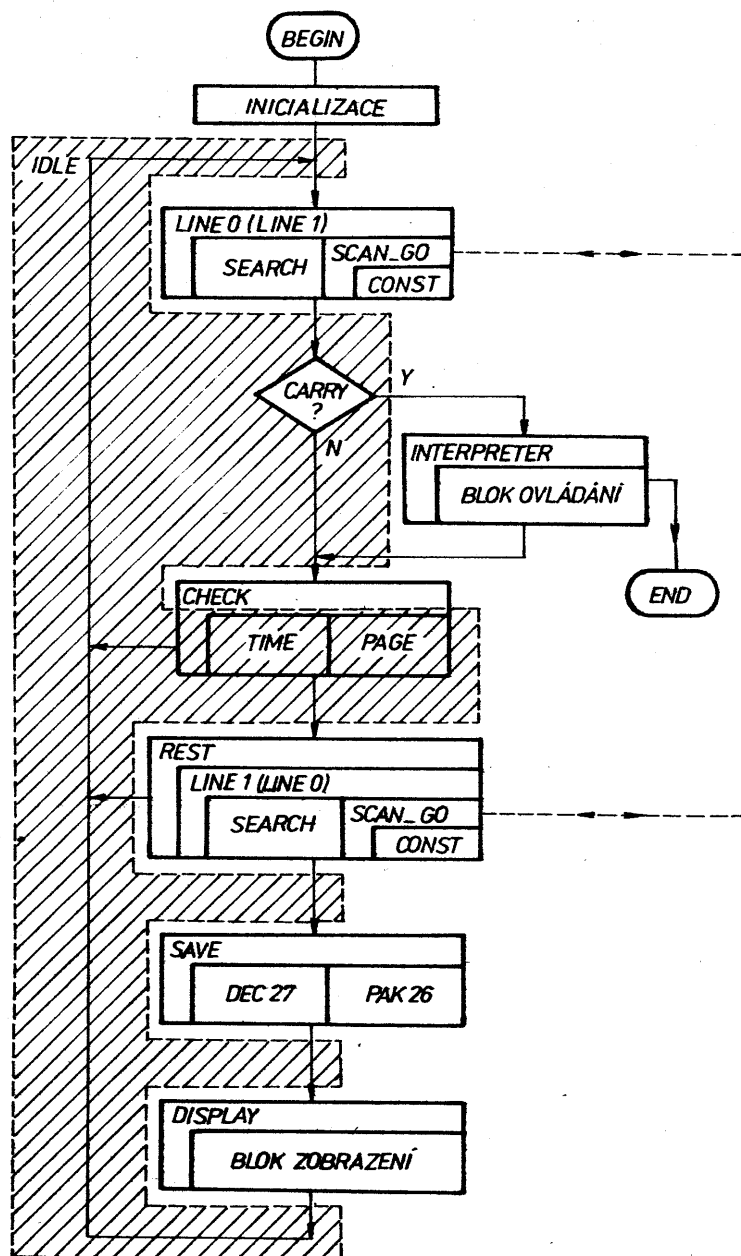
Základní požadavky na činnost programu TELETEXT byly stanoveny srovnáním se standardními dekodéry teletextu.

V první řadě je třeba, aby program umožnil plnohodnotné zobrazování vysílaných stránek, tj. aby správně reagoval na přenášené řídicí kódy a kódy alfanumerických a grafických znaků. Dále musí správně vyhodnocovat informace, obsažené v doplňkových pake- tech.

Za druhé je nutné, aby uživatelé poskytli stejný nebo takřka stejný kom-

fort při ovládání zvláštních funkcí dekodéru. Tyto funkce jsou popsány dále.

Obecné možnosti grafického zpracování teletextové stránky vyplývají z článku „TELETEXT — popis...“. Program je schopen z přijatých informací tuto stránku věrně zobrazit včetně zobrazení znaků národních abeced. Výjimku tvoří ty části stránky, kde je použita funkce blikání (odst. 4.2.9 čl. „TELETEXT — popis...“). Uvedené části budou zobrazovány tak, jako by tato funkce nebyla použita. Dále může u určitých částí stránky, vytvořených pomocí grafických znaků, docházet k nepřesnostem v barevném podání. Tyto chyby v zobrazení vznikají při překrytí rastru teletextové stránky (24 řádků po 40 pozicích pro znaky) s rastrem mikropočítače Sord a Spectrum (24 řádků po 32 pozicích pro znaky). U mikropočítače Sharp k těmto chybám nedochází.



25. doplňkový řádek, který zobrazují standardní dekodéry pod kompletní stránkou, je možné vyvolat stisknutím klávesy <SP>. Přítomnost tohoto řádku indikuje znak „>“ v levém horním rohu obrazu. Doplňkový řádek se zobrazí na místě záhlaví stránky.

Vzhledem k tomu, že videesignál z mikropočítače není synchronizován s přijímaným televizním videesignálem, není možné ani při využití vhodného přepínače realizovat ty funkce, běžné u standardních dekodérů, které vyžadují vložení (příp. prolnutí) teletextové stránky nebo její části do televizního obrazu. To se týká např. vložení časového údaje, titulkování pořadů, vložení tzv. zpravodajského vstupu, prolnutí teletextové stránky do tv obrazu, využití funkce „budík“ atd. Při příjmu s využitím dvou televizorů je možné na jednom zobrazovat televizní pořad, na druhém se pak objevují příslušné titulky. Obdobně je možné nepřímo realizovat i funkci „budík“.

Základní funkcí u každého dekodéru je funkce, která umožňuje zvolit požadovanou stránku. Stránka se volí za-

| Vývod obvodu 8255A | Signál | Signál (Centronics) |
|--------------------|--------|---------------------|
| 18 | PB 0 | Data 0 |
| 19 | PB 1 | Data 1 |
| 20 | PB 3 | Data 2 |
| 21 | PB 3 | Data 3 |
| 22 | PB 4 | Data 4 |
| 23 | PB 5 | Data 5 |
| 24 | PB 6 | Data 6 |
| 25 | PB 7 | Data 7 |
| 15 | PC 1 | BUSY |
| 11 | PC 6 | STR |
| 07 | GROUND | GROUND |

Obr. 2. Připojení tiskárny k mikropočítači ZX Spectrum

dáním třiciferného čísla (tj. včetně čísla magazínu). Tuto funkci samozřejmě program umožňuje a to kdykoli, pokud se nachází v základním režimu (tj. nejsou aplikovány funkce HOLD, DOUBLE nebo FAST SEARCH, viz dále). Jednotlivé číslice se objevují postupně v levém horním rohu obrazu vedle

písmene „P“ nebo znaku „>“. Pokud dojde k chybě při zadávání, stačí doplnit zbývající pozice libovolnými číslicemi, stisknout <CR> a provést novou volbu. Na rozdíl od standardních dekodérů vyžaduje program potvrzení volby klávesou <CR>. Po tomto potvrzení se vedle zvoleného čísla stránky objeví měnič se čísla právě přijatých stránek. Přitom nejsou zobrazována ta čísla stránek, která nenásledují v postupném číselném pořadí. Jsou to většinou přehledové stránky, které se vysílají několikrát v jednom cyklu, nebo stránky, přenášející titulky. Jakmile dojde k nalezení stránky se stejným číslem jako je zadáno, čísla přijímaných stránek zmizí a zobrazí se požadovaná stránka. Jestliže je tato stránka znehodnocena rušením, je možné provést opakované načtení klávesou „A“. Případné rotující podstránky se načítají automaticky.

Program také umožňuje zrychlenou volbu pomocí tzv. sdružených stránek. Tato možnost je indikována znakem „>“ v levém horním rohu obrazu. Po zobrazení 25. řádku klávesou <SP> je možné provést klávesou „1“ až „4“ zrychlenou volbu podle přehledu v 25. řádku.

Pomocí klávesy „I“ je možné přímo volit tzv. indexovou stránku, tj. stránku se základním přehledem.

Další běžnou funkcí je zobrazení reálného času. Čas bývá zobrazován na posledních osmi pozicích prvního řádku (záhlaví) nejčastěji ve tvaru 00.00.00. Program umožňuje průběžné zobrazení času, pouze při načtení stránky se čas zastaví na dobu potřebnou pro zpracování informací (řádově stovky ms). Zobrazení času je potlačeno při využití zvláštních funkcí DOUBLE a TIME CODE.

Pod označením zvláštní nebo doplňkové jsou u standardních dekodérů běžné následující funkce:

— Dvojitá výška (DOUBLE) — umožňuje pro lepší rozlišení zobrazit horní a dolní polovinu zvolené stránky v dvojnásobné výšce.

— Podržení rotující podstránky (HOLD) — pokud jsou podstránky zobrazovány po příliš krátkou dobu, je možné touto funkcí jejich rotaci přerušit a podržet zobrazovanou podstránku na požadovanou dobu. Tuto funkci je možné využít i u normálních stránek, a to tehdy, když vlivem rušení dochází k častému přepisování načtené stránky stránkou falešnou.

— Zobrazení skrytých znaků (REVEAL) — tato funkce umožňuje zviditelnění znaků, které jsou při prvním zobrazení skryty (zobrazeny jako mezerky). Využívá se především pro skrytí odpovědí u kvízů nebo pro skrytí informací, určených pouze pro některé uživatele.

— Přímá volba podstránky (TIME CODE) — pomocí této funkce je možné zvolit číslo požadované podstránky obdobně jako číslo základní stránky (volba je možná pouze u některých podstránek, tato možnost bývá vyznačena). Volba se provádí po zvolení čísla stránky. Číslo podstránky se většinou zobrazí na místě reálného času.

Tímto způsobem se zadává i čas pro aplikaci funkce „budík“. Stránka, na které je budík zobrazen, je vlastně složena z rotujících podstránek, jejichž

čísla odpovídají reálnému času. To znamená, že tato podstránka se zobrazí v době, kdy se její číslo bude shodovat s nastaveným časem.

Program umožňuje realizaci všech těchto funkcí. Jejich přesné použití je popsáno v části Popis ovládání.

Program navíc umožňuje vytisknout zvolenou stránku dvěma způsoby. U tiskárny se předpokládá rozhraní Centronics, nejlépe standard IBM nebo Epson. Nastavení způsobu tisku a vlastní tisk se provede příkazy, které jsou dostupné po zobrazení menu (viz část Popis ovládání). Tabulka propojení tiskárny a interfejsu s obvodem 8255A pro mikropočítač ZX Spectrum je na obr. 2.

Popis programu TELETXT

Programové vybavení bylo vytvořeno pro mikroprocesor Z80 a implementováno na mikropočítači Sord M5, Sharp MZ 800 a ZX Spectrum. Program je napsán v makroassembleru Z80, zdrojový text má délku asi 130 kB. Přeložený strojový kód má délku asi 14 kB, z toho je asi 6 kB vlastní program, zbytek tvoří tabulky a data pro dekódování.

Celkový vývojový diagram programu je na str. 17. Pro přehlednost jsou zakresleny pouze hlavní vazby mezi jednotlivými bloky.

Hlavní částí programu TELETXT je procedura *IDLE*, která zajišťuje vzájemné propojení a volání jednotlivých bloků programu a dále zajišťuje i některé stále se opakující funkce (bloky *TIME*, *PAGE*).

Stručný popis funkcí jednotlivých bloků:

INICIALIZACE — základní inicializace systému.

LINE0, LINE1 — vyhledává a dekóduje adresu řádku, načítá záhlaví zvolené stránky.

SEARCH — vyhledává rámcový (framing) kód.

SCAN-GO — čeká na ukončení nejbližšího signálu *GO* a současně pomocí bloku *CONST* testuje klávesnici na stisk klávesy. Po ukončení *GO* přesune data z vyrovnávací paměti adaptéru do operační paměti mikropočítače pro další zpracování.

CONST — testuje klávesnici.

CARRY — test vložky indikující stisknutí klávesy.

INTERPRETER — volá příslušnou rutinu z BLOKU OVLÁDÁNÍ.

BLOK OVLÁDÁNÍ — obsahuje rutiny nutné pro provádění příkazů uživatele (*DOUBLE*, *HOLD*, *REVEAL*, *CLOCK*, *TIME*, *CODE*, *FAST*, *SEARCH*, *INDEX*, *MENU*, *PRINT*, *HELP*, *SETUP*, *EXIT*).

CHECK — volá příslušnou rutinu *TIME* a *PAGE*.

TIME — porovnává systémový čas s údajem o reálném čase, přenášeném v záhlavích stránek.

PAGE — při vyhledávání zvolené stránky porovnává číslo právě načtené a zvolené stránky (včetně čísla magazínu).

REST — načítá zbytek stránky až po hlavičku následující stránky.

SAVE — volá rutiny nutné pro zpracování paketu 27 a paketu 26.

DEK27 — dekóduje paket 27, **PAK26** — dekóduje paket 26.

DISPLAY — řídí zobrazování stránky, volá potřebné rutiny z BLOKU ZOBRAZENÍ.

BLOK ZOBRAZENÍ — obsahuje rutiny, nutné pro správné zobrazení stránky.

Popis některých vybraných částí programu

ÚVODNÍ BLOK

V úvodním bloku jsou definovány nutné makroinstrukce pro komunikaci s paralelním rozhraním a velikosti jednotlivých bafrů. Je uveden příklad minimálních definic, potřebných pro činnost základní části programu.

```
*****
*                               *
*   Úvodní blok                *
*                               *
*****

FRAMING EQU    ""             ; Framingový kód
ENABLE0 EQU    00001110B      ; ENAB E=0
ENABLE1 EQU    00001111B      ; ENAB E=1

;
;   PROGRAMOVÁNÍ 8255
;
;   1. Nastavení režimu
;   Bit   7   6   5   4   3   2   1   0
;   :       :       :       :       :       :
;   :       :       :       :       :       : Port C (lower) 1-Input, 0-Output
;   :       :       :       :       :       : Port B 1-Input, 0-Output
;   :       :       :       :       :       : Mode selection 0-Mode 0, 1-Mode 1
;   :       :       :       :       :       : Port C (upper) 1-Input, 0-Output
;   :       :       :       :       :       : Port A 1-Input, 0-Output
;   :       :       :       :       :       : Mode selection 00-Mode 0, 01-Mode 1, 1X-Mode 2
;   :       :       :       :       :       : Mode set flag 1-Active
;
;   2. Bit set/reset format
;   Bit   7   6   5   4   3   2   1   0
;   :       :       :       :       :       :
;   :       :       :       :       :       : Bit set/reset flag 1-Set, 0-Reset
;   :       :       :       :       :       : Bit select 000-0 : 111-7
;   :       :       :       :       :       : Unused
;   :       :       :       :       :       : Bit set/reset flag 0-Active

CW EQU 10010011B             ; Ridici slovo pro 8255
; PA - vstup dat z dekoderu
; PB - nepoužit-nastaven jako vstup
; PC0 - /GO signal z dekoderu
; PC7 - ENABLE signal do dekoderu
; Velikost bufferu pro PIO ( max. pro
; 8 ttx. datových radku v pulsníku )

PIO SIZE EQU 448
```

VYHLEDÁNÍ A DEKÓDOVÁNÍ ŘÁDKU

V další části je popis hlavní procedury pro vyčítání dat z adaptéru a pro úpravu jednoho teletextového řádku. Procedura volá podprogramy nižší úrovně (viz dále), které jsou zčásti závislé na hardware počítače. Vlastní hlavní procedura pro vyčítání je volána z hlavní programové smyčky (procedura *IDLE*).

```
*****
*                               *
*   Vyhledání a dekódování radku *
*                               *
*****

;
;   Program vyhledá a dekóduje adresu radku do oblasti ROWADR
;   Je-li to radek 00 ( hlavička ), provede dekódování do oblasti HEAD
;   pres rutiny :
;   SEARCH.....Vyhledání framingového kodu
;   SCAN_GO.....Načtení dat z teletextu
;   DECODE.....Makro, které dekóduje Hammingovy kody
;   Vstup: F(CY)=1 - bylo prerušeno čtení
;   :       jinak F(CY)=0
;   :       HL - Adresa prvního datového bytu ( za záhlavím )
;
;   Vstup: LINE1 - HL, BC a PIOBUF nastaveny z předchozího volání
;   :       LINE0 - Načítá data do PIOBUF a nastavuje HL, BC

LINE0:
CALL SCAN_GO
RET C
LD HL, PIOBUF
LD BC, PIO SIZE
LINE1: CALL SEARCH
JR NZ, LINE0
LD A, (HL)
DECODE
JP M, LINE1
LD E, A
INC HL
LD A, (HL)
DECODE
JP M, LINE1
INC HL
SLA A
BIT 3, E
JR Z, LINE4
INC A
LINE4: LD (ROWADR), A
LD D, A
LD A, E
AND 111B
```

LINE2:

Následující rutina je používána pro vyhledání začátků textových řádků. Začátek řádku je identifikován přítomností rámcového kódu (framing code). Jestliže následují dva tyto kódy za sebou, považuje se za platný druhý výskyt.

```

*****
;*                                     *
;*      Vyhledavani Framingova kodu      *
;*                                     *
*****
;      Program vyhledava framinguv kod od adresy (HL) v delce (BC)
;      a testuje pretečení z minulého vyhledavani.
;      Vstup: Z=1 ... Nasel framing HL addr(Framing1)

```

```

SEARCH:  BIT    7,B           ; Test na pretečení rozsahu
          RET     NZ           ; Framing nenalezen-pretek!
          LD      A,FRAMING    ; Co hledáme
          CPIR                     ; Hledej
          RET     NZ           ; Navrát - nenalezen
          CP      (HL)         ; Test na 2* framing za sebou
          JR      Z,SEARCH1     ; Jsou 2 framings
          XOR     A             ; Z=1
          RET                     ; Navrát Z
SEARCH1: INC     HL            ; Preskok první framing
          DEC     BC            ; Oprav počítadlo
          XOR     A             ; Z=1
          RET                     ; Navrát Z

```

Dále uvedená tabulka umožňuje velmi rychlé dekódování a vyhodnocení (případně opravení) Hammingova kódu (varianta A, viz „TELETEXT — popis systému ...“). Tuto tabulku využívá makro DECODE volané v LINE0 a v LINE1.

```

*****
;*
;*   Prekodovací TURBO tabulka
;*
*****
;*
;*   Tabulka pro TURBO dekodování
;*   (c) 1988 Pavel BRYCHTA

```

```

RADIX 16
DB 0FF,00C,001,0FF,00A,0FF,0FF,009
DB 00A,0FF,0FF,00B,00A,00A,00A,0FF
DB 00B,0FF,0FF,00B,0FF,000,00D,0FF
DB 0FF,00B,00B,00B,00A,0FF,0FF,00B
DB 00C,00C,0FF,00C,0FF,00C,00D,0FF
DB 0FF,00C,0FF,0FF,00A,0FF,0FF,007
DB 0FF,00C,00D,0FF,00D,0FF,00D,00D
DB 006,0FF,0FF,00B,0FF,00E,00D,0FF
DB 0FF,0FF,0FF,009,0FF,009,009,009
DB 0FF,002,00F,0FF,00A,0FF,0FF,009
DB 00B,00B,00B,0FF,00B,0FF,0FF,009
DB 00B,0FF,0FF,00B,0FF,00E,003,0FF
DB 0FF,00C,0FF,0FF,00A,0FF,0FF,009
DB 00F,0FF,00F,00F,0FF,00E,00F,0FF
DB 00B,0FF,0FF,005,0FF,00E,00D,0FF
DB 0FF,00E,00F,0FF,00E,00E,0FF,00E
DB 001,0FF,001,001,0FF,000,001,0FF
DB 0FF,002,001,0FF,00A,0FF,0FF,007
DB 0FF,000,001,0FF,000,000,0FF,000
DB 006,0FF,0FF,00B,0FF,000,003,0FF
DB 0FF,00C,001,0FF,004,0FF,0FF,007
DB 006,0FF,0FF,007,0FF,007,007,007
DB 006,0FF,0FF,005,0FF,000,00D,0FF
DB 006,006,006,0FF,006,0FF,0FF,007
DB 0FF,002,001,0FF,00A,0FF,0FF,009
DB 002,002,0FF,002,0FF,002,003,0FF
DB 008,0FF,0FF,005,0FF,000,003,0FF
DB 0FF,002,003,0FF,003,0FF,003,003
DB 004,0FF,0FF,005,004,004,004,0FF
DB 0FF,002,0FF,004,0FF,0FF,007
DB 0FF,005,005,005,004,0FF,0FF,005
DB 006,0FF,0FF,005,0FF,00E,003,0FF
RADIX 0AH

```

```

SCANLO:
CALL    CONST          ; Get console status
OR      A              ; Nastav vlnajku Z/NZ
SCF     A              ; Pripadne vlnajku INTERRUPT
RET     NZ             ; Byla stisknuta klavesa
IN      A,(PIOC)       ; Nacti status GO signalu
AND     1              ; Maskuj pouze tento bit
JR      NZ,SCANLO     ; GO jeste neprislo

GO_SCAN:
IN      A,(PIOC)       ; Test na jeste trvani GO
AND     1              ; Maskuj GO bit
JR      Z,GO_SCAN     ; GO jeste trva
LD      A,ENABL1       ; Povoline vycitani dat z dekoderu
OUT     (CTRL),A       ; pres PC7
LD      HL,PIORBUF     ; Kam budeme ukladat data
LD      BC,PIDA        ; Odkud budeme brat + pocitadlo
INIR    A              ; Nacteme prvni 256 bytu
LD      BC,256*(PIOSIZE-256)+PIDA; Nacteme zbytek dat
TNR     A              ;
LD      A,ENABL0       ; Zakazeme vycitani dat z dekoderu
OUT     (CTRL),A       ; pres PC7
XOR     A              ; Nuluji CY flag
RET

```

| TABLE: | DB | 001,0FF,001,001,0FF,000,001,0FF |
|--------|----|---------------------------------|
| | DB | 0FF,002,001,0FF,00A,0FF,0FF,007 |
| | DB | 0FF,000,001,0FF,000,000,0FF,000 |
| | DB | 006,0FF,0FF,008,0FF,000,003,0FF |
| | DB | 0FF,00C,001,0FF,004,0FF,0FF,007 |
| | DB | 006,0FF,0FF,007,0FF,007,007,007 |
| | DB | 006,0FF,0FF,005,0FF,000,000,0FF |
| | DB | 006,006,006,0FF,006,0FF,0FF,007 |
| | DB | 0FF,002,001,0FF,004,0FF,0FF,009 |
| | DB | 002,002,0FF,002,0FF,002,003,0FF |
| | DB | 008,0FF,0FF,005,0FF,000,003,0FF |
| | DB | 9FF,002,003,0FF,003,0FF,003,003 |
| | DB | 004,0FF,0FF,005,004,004,004,0FF |
| | DB | 0FF,002,00F,0FF,004,0FF,0FF,007 |
| | DB | 0FF,005,005,005,004,0FF,0FF,005 |
| | DB | 006,0FF,0FF,005,0FF,00E,003,0FF |

.RADIX 0AH

Popis ovládání

Po zavedení programu z kazety (diskety) se zobrazí titulní strana a hlavní menu. Menu obsahuje tyto tři základní varianty:

Setup Print Help

Jednotlivé varianty se volí klávesou odpovídající zvýrazněnému písmenu.

Setup — umožňuje nastavení způsobu vysílání řídicích znaků na tiskárnu ve formě CR nebo CR+LF a následující způsoby kopie obrazovky:

Normal — normální kopie — obsahuje pouze platné ASCII znaky, grafika se zobrazí pomocí odpovídajících ASCII znaků, znaky národních variant jsou zobrazeny bez diakritických znamének.

Shaded — stínovaná grafická kopie — využívá možností tiskáren standardu IBM a Epson.

Print — provede vytištění načtené stránky podle podmínek, nastavených v předchozím bodě.

Help — zobrazí seznam písmenových příkazů pro standardní ovládání. Jsou to tyto příkazy:

Q-9 — **page#** — volba stránky — provede se zadáním třiciferného čísla, po kterém následuje <CR>.

A — **again** — znovu se provede načtení zvolené strany.

C — **clock** — po předchozí přímé volbě podstrany se zobrazí potlačený reálný čas.

D — **double** — horní a dolní polovina načtené stránky se postupně zobrazí v dvojnásobné výšce. Rotování se provádí libovolnou klávesou.

E — **exit** — návrat do operačního systému mikropočítače.

H — **hold** — podržení rotující podstránky až do dalšího stisku libovolné klávesy. Funkce je indikována písmenem „H“ v levém horním rohu obrazu.

I — **index** — automatická volba indexové stránky (stránka 100, případně stránka nastavená pomocí paketu 27).

M — **menu** — zobrazí se menu.

R — **reveal** — zobrazí se znaky, které jsou při načtení stránky skryty.

T — **time code** — přímá volba podstránky — umožňuje zadat čtyřciferné číslo podstránky. Po zadání čísla stránky stiskneme „T“ (na místě reálného času se zobrazí Time0000), zadáme číslo podstránky a stiskneme <CR>.

<SP> — **fast search** — volba sdružených stránek. V případě, že se přenáší adresy tzv. sdružených stránek, které jsou určeny k automatickému ukládání do paměti dekodéru, je jejich přítomnost indikována znakem „>“ v levém horním rohu obrazu. Pak je možné pomocí klávesy <SP> vyvolat zobrazení doplňkového 25. řádku, přenášejícího přehled sdružených stránek. Volba sdružené stránky se provede stisknutím odpovídající klávesy „1“ až „4“. Klávesy jsou přiřazeny barevným skupinám následujícím způsobem:

| | |
|-------------|---------------|
| klávesa „1“ | „červená“ |
| klávesa „2“ | „zelená“ |
| klávesa „3“ | „žlutá“ |
| klávesa „4“ | „modrozelená“ |

Publikovat celý výpis programu pro příjem teletextu nebylo vzhledem k jeho rozsahu reálné. Proto jsme zařídili v rámci projektu MIKROBÁZE, aby nahrávku programu na kazetě dodávala 602. ZO Svazarmu v Praze. Zatím jsou k dispozici programy pro počítače ZX-Spectrum, SORD M5, Sharp MZ800, připravují se další včetně IBM-PC. O program pro svůj počítač si tedy napište na adresu: 602. ZO Svazarmu, ul. dr. Z. Wintra 8, 160 00 Praha 6.

Dlouhý spánek Gajany

Carola Biedermannová

Za jasných nocí bliká na obloze malá hvězdička. Ze Země není vidět, že kolem ní krouží tři planety. Dvě jsou mrtvé. Hromady kamení, mrazivé pustiny. Třetí je živá. Jmenuje se Gajana.

Podnebí Gajany je teplé a kromě oblasti tropických dešťových pralesů suché. Rána jsou chladná a svěží, poledne horká do úmoru a večery dlouhé, vlahé, soumravné. Je to mírná planeta. Čas plyne tiše a nemá velký význam.

Na Zemi vesele skotačili brontosaurové a jiné těžkotonážní potvory, které byly později odkázány do říše pohádek, aby ještě později byly s úctou a pietně objevovány jejich kosti.

Na Gajaně slezl podnikavý opičák nebo zvíře opici podobné ze stromu a postavil se na zadní končetiny, aby se podíval do dálky nad vysokou travou gajanské stepi. V trávě

uviděl pasoucí se zvíře podobné králíkovi. Napadlo ho vzít do ruky klacek a králíka praštit. Králík padl s roztrášenou lebkou.

Opičák si dřepel vedle králíčí mrtvolky a tupě na ni zíral. Přitom mimoděk olízl klacek se zbytky králíčího mozku. Chutnal podobně jako stromoví červi, kterými se živil.

Na planetě Zemi slezl první podnikavý opičák ze stromu v době, kdy lidé na Gajaně podnikali pokusy s chovem zvířat a pěstováním rostlin. Když si na planetě Zemi lidé vzájemně kradli nasbírané rostliny a ochočená zvířata, na Gajaně již stála bílá města. Vysoké ženy s lehce namodralou pletí a korunou zlatých vlasů a muži, do pasu nazí a v dlouhých bílých sukních procházeli stinnými podloubími ulic kolem plně automatizovaných obchodů a pracovali v nekonečně bělostných klimatizovaných sálech a chodbách počítačů. Za noci hleděli ke hvězdám.

Podnebí Gajany je suché a teplé, déšť nezavírá lidi v jejich přibytých. Lidé nezapomněli na široké stepi, ze kterých přišli. Ve volných dnech odcházeli do malých domků ve stepi, aby žili blíž přírodě. Pěstovali ovoce a zeleninu, žili bez

jídel, která se sama uvaří a bez vozidel, která se sama řídí. Volných dnů přibývalo, počítače ve městech byly čím dál tím schopnější a samostatnější. Lidé Gajany se postupně vraceli do svých stepí. Počítače pro ně vytvořily rozvázkovou službu. Ale zásobovací vozy se vracely plné, lidé si pěstovali potraviny vlastníma rukama a tkali látky na primitivních stavech. Nechtěli už trápit děti školou, vzděláním, prací. Mysleli to dobře, když jim chtěli dopřát svobodu. Tehdy.

Podnebí Gajany je teplé, suché a mírné. Jitra jsou tichá a osvěžující. Lidé Gajany neznali válčení, oblíbenou zábavu lidí na Zemi. Stádní pud u něj zanikal. Ve stepích bylo místa dost.

Bílá města Gajany osiřela.

Počítače postupně rušily výrobu věcí, které nikdo nepotřeboval. Automaty a roboty udržovaly opuštěná města nebo čekaly, pečlivě nakonzervovány, ve skladištích. Nabýskaná vozidla stála v garážích, obchody byly prázdné, jen stepní vítr se proháněl podlouhými. V dlouhých chodbách počítačových center blikala světélka a svítily obrazovky. Počítače čekaly na muže v dlouhých sukních se zlatými vlasy k ramenům, na dívky v bílých řízách, protože, jak pravil jejich program a jejich zkušenost, jen tým člověka s počítačem dává fungující celek. Lidé však odešli. Opustili i své domky, které stejně už neuměli pořádně udržovat, a žili v dřevěných srubech nebo zahloubených polozemnicích.

Na Zemi lidé pomalu končili s toulavým pasteveckým způsobem života a zvykali si žít v trvalejších sídlištích. Bude trvat ještě dlouho, než bude postaveno město u Catal HŮyuk, než se zrodí na ostrovech první říše, po které zůstanou jen báje o Lemurii, zemi Mu či Atlantidě.

Počítače na Gajaně žily bez lidí. Měly spoustu času. Využíly ho na zdokonalení, získaly první záblesky skutečného vědomí. Bylo to vědomí osamělosti.

O údržbu počítačů se staraly roboty, slunce Gajany dodávalo do slunečních baterií neomezené množství energie. Ale přicházel stesk. Lidé zapomínali na počítače a na bílá města, ze kterých odešli do širých stepí za svobodou. Města se v jejich vyprávnění stávala podivným přetechizovaným peklem. Po lidech zůstaly linky telefonů, stanice bezdrátového spojení. Počítače je měly na dosah. A tak se propojily do jednoho planetárního systému. S nadějí, že přemohou samotu. Vznikl velký počítač Gajany.

Na Gajaně znamená čas málo a pro počítač ještě méně. — Jen samota odkapává v drobných krupějkách a měří dobu čekání. Velká kaluž samoty znamená stesk. Z kaluží vznikají potůčky, jezírka... Počítač Gajany se odhodlal k činu. Poslal za lidmi roboty, aby je přivedly zpátky, když bílá města čekají a počítač je stále připraven jim sloužit.

Lidé Gajany již zapomněli umění práce s kovy. Táhl se svými stády od pastviny k pastvině, bydleli ve stanech z páchnoucích zvířecích kůže, pojídali napul syrové maso a divoké rostliny. Večer se u táborových ohnů vyprávěly pověsti o bílém pekle měst. Nedůvěřivě, se strachem hleděli lidé na roboty, které jim neubližovaly a něco říkaly. Jistě to byli démoni pekla.

Řeč lidí se změnila. Bylo v ní málo slov a zněla jinak, než bylo uloženo v paměti velkého počítače. Počítač měl v té době bohatší slovník než nejchytřejší z lidí. Možná, že lidé na Gajaně se chvíli pokoušeli pochopit, co od nich ty lesklé krabice chtějí, než je poslali pryč nevybíravými slovy. Neměli ostatně z čeho vybírat a měli spoustu jiných starostí. Hodně se jich rodilo, ještě více umíralo, pořád řádily nějaké nemoci, bolely je zuby, pokud nějaké měli, dobytek se plašil a mouchy otravovaly.

Velký počítač Gajany usoudil vcelku správně, že lidé se nevrátí, že zůstal sám. Hledal řeší.

Přestal udržovat zbytečná města, ta se po čase zhroutila v hromady sutin a z těch pomalu vznikala písek. Zůstala jen počítačová centra, sluneční baterie, výroba obslužných robotů a hvězdárny.

Na Gajaně čas mnoho neznamenal a pro počítač ještě méně. Počítač hleděl k hvězdné obloze. Počítal, přemýšlel, možná i snil. Byl pánem planety a myslel na hvězdy.

I na Zemi lidé hleděli ke hvězdám, dávali jim jména svých bohů, dávali jména souhvězdím. Jedno dostalo jméno Pastýř.

Stepní vítr rozfoukal po Gajaně písek, který zbyl z bílých měst. Přibývalo písečných bouří, ve kterých hynuli lidé i zvířata. Stepi vysychaly pod vrstvou vápenatého písku, měnily se v poušť. Lidé se přestěhovali do jeskyní a živili se lovem drobné zvěře. Usidlovali se na okrajích pralesů, které poušť nemohla pohltit, neměla na to dost síly.

Jeden podnikavý muž, drobný a obratný, vylezl na strom. Objevil hnízdo stromových červů, ochutnal a ejhle, podobalo se to chuti vzácného králičího mozečku. Navíc našel několik divokých plodů, prostě měl šťastný den. Po dlouhé době se dosyta najedl.

Lidé z Gajany se začali stěhovat do pralesů. Postavili si hnízda z větví a bydleli na stromech. Prales se dočkal.

Počítač vyrobil sérii hlídkovacích zařízení, továrna na výrobu obslužných robotů to hladce zvládla a pouště Gajany byly ideálním letištěm. Zařízení byla prakticky nezničitelná, v neměnném kosmickém vzduchoprázdnu mohla čekat neomezenou dobu. Vypočítal vhodná místa — měl k dispozici hvězdárny celé Gajany — a rozeslal své hlídky do vesmíru.

Lidé Gajany nemohli ze svých lesních skryší zahlédnout světélka, ztrácející se mezi hvězdami. Velký počítač Gajany se ponořil do dlouhého čekání. Možná i snil. A zatím poušť zakryla poslední zbytky civilizace, poušť se srdcem z kovu a polovodičů, kterou daleko od sebe probleskovaly povrchy slunečních baterií...

Na Zemi Atlantis skončila v rozbouřených živlech. Harrapa a Mohendžoro prošly časem, sumerské paláce pokryl prach a hnila. Jako brontosauři, i říše Chetitů byla vykážána do pověstí, aby po dlouhém čase byla archeology pietně odkryta. Egypťané postavili své pyramidy, pro které čas mnoho neznamenal. Kréťané ovládali moře a jako sport pěstovali skok přes býka, dědictví bezejmenného města od Catal HŮyuk. Řekové postavili své bílé chrámy, které časem Turci opět zbořili, a tesali sochy, které zůstanou měřítkem krásy v době, kdy jedna ze sond Gajany začne plnit svůj program.

V dešťových pralesích Gajany potomci lidí sestupovali na zem jen zřídka. V tom případě se pohybovali po čtyřech chápavých končetinách. Jednomu druhu se vyvinul chápavý ohon, což se pro život na stromech bezesporu hodilo. Děsili se otevřeného prostoru, za kterým odešli ze svých měst a který se stal pouští. Vzpomínka na dávnou civilizaci v nich vyhasla. Jejich slovník se skládal z posunků, skřeků, kňourání a očichávání.

Na Zemi začala a skončila doba křesťanského temna. Učení mnichové Campanella, Bruno a Kepler a po nich učení laikové Galileo a Herschel obraceli zrak lidstva ke hvězdám, což nebylo právě snadné. Lidstvo se raději bavilo válčením a jinými zmatky. Byl vynalezen parní stroj, objevena elektřina, pracoval Edison a Marconi. Byl objeven bezdrátový přenos zpráv. Vesmírný posel začal plnit program.

V roce 1927 podle pozemského počítání času byl již v éteru slušný provoz. Sonda vysílala na Gajanu všechno, co zachytíla, ostatní bylo starostí velkého počítače. Určitě s tím měl starosti dost. Sonda zatím plnila svůj druhý úkol.

Potomci lidí Gajany obrostli zlatou srstí a vesele skákali z větve na větev, aniž by pocítili nespokojenost se svým osudem, když Taylor a Young zaznamenali zpoždění odrazu radiových signálů. V roce 1928 se totéž povedlo Halsovi a Strömerovi, jev podrobně prozkoumal Van der Pol, který zjistil zpoždění ozvěn od tří do patnácti sekund a pozměněné trvání některých signálů, zatímco frekvence vždy souhlasila.

Jenže lidstvo mělo jiné starosti.

V roce 1960 oživil problém zpožděného odrazu Bracewell v časopisu Nature. V roce 1965 si J. Strong uvědomil, že místo odrazu je současně jedním z liberálních — rovnovážných bodů soustavy Země — Měsíc a tedy nejlogičtější místem například pro umístění meziplanetární sondy. V roce 1973 napadlo Lunana přenést číselnou řadu zpožděných ozvěn do grafu. Před jeho mírně udiveným zrakem se objevila mapa souhvězdí Pastýře, jak by se jevila při pohledu ze Země před třinácti tisíci léty.

Jenže souhvězdí Pastýře je pro lidi Země trochu z ruky a lidé mají spoustu jiných starostí a málo času. Na Zemi čas znamená hodně.

V roce 2020 lidé opět jednou vesele válčili, pro změnu o kousek území planety Mars, které v podstatě k ničemu nebylo. Znechucený pilot bojové kosmické lodi se vracel na základnu. Šarvátka se protáhla, zmeškal randa s bezvadnou kočkou, byl unavený, dostal vynadáno a zaražené vycházky. Vztek střílel po všem, co mu přišlo na mušku. Pokaždé trefil.

Ze zasaženého meteoru vyšlehl prudký záblesk, jehož směr pilot nezaznamenal. Fuj, to jsem se lekl, bylo všechno, co ho napadlo.

V pouštích Gajany čas nic neznamenal, poušť neměnil svou zlatožlutou tvář. V tropických dešťových pralesích svobodně skotačil několik opic. V písku pouští občas probleskne povrch sluneční baterie. Skryt pod pískem snil počítač své sny.

V soustavě dvou těles, například Země—Měsíc, je pět libračních bodů.

CP/M, RAMDISK A ŘADIČ PRUŽNÉHO DISKU

PRO MIKROPOČÍTAČ PROGRAMOVĚ KOMPATIBILNÍ SE ZX SPECTRUM

Ing. Aleš JUŘÍK, FE VUT, Božetěchova 2, 612 00 Brno

V současné době je již známo několik úprav mikropočítačů ZX Spectrum pro práci pod operačním systémem CP/M. Tento článek je věnován nejen těm, kteří si postavili mikropočítač kompatibilní se ZX Spectrum podle [1], ale také těm, kteří chtějí provozovat na svém osmibitovém mikropočítači operační systém CP/M, který je v současném počítačovém světě bezesporu nejrozšířenějším operačním systémem, orientovaným na osmibitové mikropočítače.

Vzhledem k tomu, že existuje již několik verzí tohoto operačního systému, je navrhovaná verze upravena tak, aby pod systémem mohly pracovat všechny programy, dostupné v tuzemsku i v zahraničí, používající důsledně modulu BIOS. Menší problémy vytváří pouze výstup na obrazovku (konzolu), protože u mikropočítače ZX Spectrum nelze dosáhnout čitelného zobrazení více než 64 znaků na řádek (nikoli standardních 80). Zato zde lze použít speciální grafické funkce, které je možno implementovat do modulu BIOS a při psaní vlastních programů využívat grafický výstup na obrazovku. Tento způsob byl vyzkoušen několika programy a poskytuje rozmanité možnosti při tvorbě programů pod operačním systémem CP/M.

Obecný popis systému CP/M nebude v tomto článku uveden, zájemce najde podrobné informace v [2] a [3]. Kromě toho je tento operační systém dostatečně znám, takže budou uváděny jen změny proti standardnímu provedení. Článek může brát jako ucelený stavební návod, i když pro majitele počítače podle [1] na něj tak lze pohlížet. Je z něj možno čerpat inspiraci pro další zájemce o operační systém CP/M, zálohovaný ramdisk nebo řadič floppydisku včetně kanálu DMA. Zároveň je možná aplikace pouze některých částí tohoto článku (tzn. CP/M jen s ramdiskem a kazetovým magnetofonem nebo CP/M jen s jednou nebo více disketovými jednotkami).

1. POPIS OBVODOVÉHO ŘEŠENÍ

Obvodové řešení vychází z faktu, že operační systém CP/M potřebuje pro svou instalaci paměťový prostor z paměti RWM přes celý základní adresový prostor mikropočítače, tj. 64 kB. Někdy se uvádí, že BIOS může být v paměti ROM. Z hlediska kompatibility s různými moduly BDOS, z nichž většina vyžaduje v oblasti BIOSu od adresy 0FB00H dolů několik desítek bajtů volného prostoru pro zásobník, je vhodné, aby celý paměťový prostor byl realizován z paměti RWM. Pro majitele mikropočítače podle [1] je tento požadavek splněn ovládním vnitřní paměti signálem SP1N a překrytím dolních 32 kB odpovídající části paměti „půjčované“ z ramdisku. Pro majitele Spectra s 80 kB vnitřní paměti RWM je nutno pouze přizpůsobit přepínání této paměti signálem SP1N tak, aby při aktivní úrovni tohoto signálu byla paměť ROM a paměť video překryta pamětí RWM. Majitelé neupraveného Spectra mohou ovládat pomocí tohoto signálu přídavnou přepínací logiku tak, aby při aktivní úrovni signálu SP1N obvod ULA měl na vstupech AB14 a AB15 úrovně logické nuly a zároveň vyblokovat vnitřní paměť ROM přivedením logické jedničky

na vstup CSROMGEN. To lze jednoduše zajistit přerušením příslušných adresových vodičů u vstupů obvodu ULA a přidáním jednoho obvodu např. typu 74LS08.

1.1 Popis obvodů ramdisku

Návrh obvodů ramdisku vychází z předpokladu, že s pamětí ramdisku se bude komunikovat jako se stránkovou systémovou pamětí, nikoli přes port instrukcemi vstupu a výstupu. Velikost stránky byla pro co nejjednodušší obvodové řešení zvolena 32 kB. Pro „půjčování“ systému (překrytí ROM a video RAM) byla zvolena stránka č. 0 (z hlediska obvodové jednoduchosti lze stejně použít i stránku s adresou nejvyšší, ale navrhovaný ramdisk může mít i menší paměťovou kapacitu).

Obvodové řešení ramdisku je na **obrázcích 1 až 4**. Umožňuje následující funkce:

1) při komunikaci s ramdiskem je příslušná třicetidvoukilobajtová stránka připojena do paměťového prostoru mikropočítače od adresy 04000H.

2) pokud se požaduje připnutí systémové paměti RWM do celého adresového prostoru mikroprocesoru, přepne se od adresy 00000H nultá stránka ramdisku. Protože při použití operačního systému CP/M není možné používat tuto stránku jako součást ramdisku a systémové paměti zároveň, je programově v BIOSu zablokováno její použití v ramdisku.

3) při aktivaci funkce BOOT (zavedení systému a jeho studený start) je do spodní části adresového prostoru přimapována paměť ROM se zaváděčem systému z diskety a do horní poloviny (od adresy 08000H) je připojena paměť RWM. Po nahrání systému program skočí do BIOSu, který odepne paměť ROM a uskutěční studený start systému. V současné době zatím modul BIOS vyžaduje přítomnost obsahu paměti ROM Spectra (využívá se z ní scroll obrazovky, funkce PLOT, DRAW a BELL). Zaváděč systému byl doplněn do volného prostoru paměti ROM.

Podle těchto požadavků dostáváme tedy signál odepínající vnitřní paměť RWM jako součet tří signálů (viz **obr. 1**).

$$SP1 = RAMDISK + (EXPRAM.A15N) + CSROM \quad (1)$$

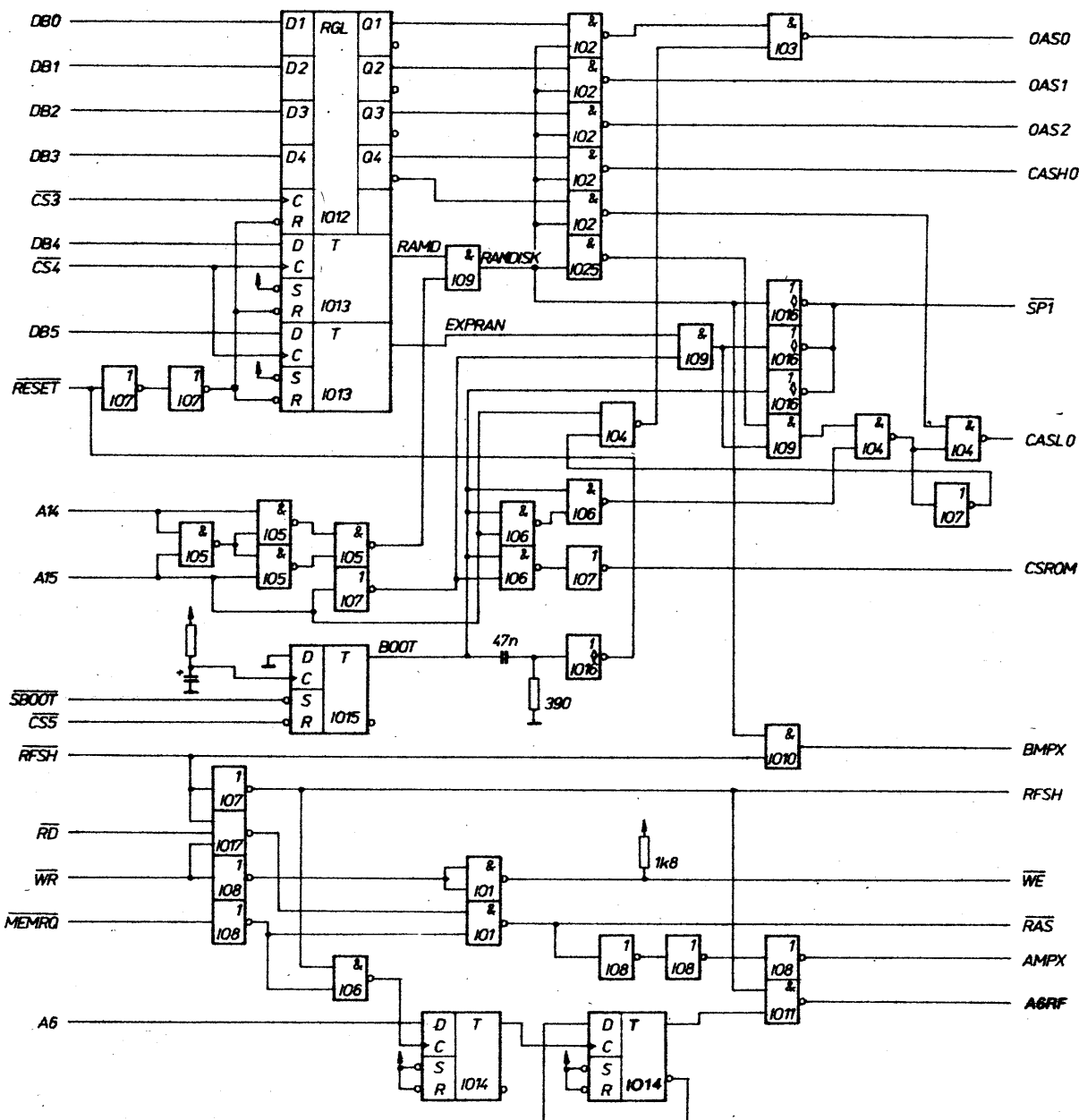
ad 1) Funkce obvodů podle tohoto bodu je zabezpečena ovládním signálu SP1N pomocí signálu RAMDISK (aktivní úroveň tohoto signálu signalizuje, že zrovna probíhá komunikace s pamětí jako s ramdiskem):

$$RAMDISK = RAMD \cdot (A14.A15N + A14N.A15) \quad (2)$$

kde signál RAMD je výstup IO13 (**obr. 1**) a signály A14 a A15 jsou příslušné bity adresové sběrnice. Do obou polovin IO13 se výběrovým signálem CS4N (instrukcí OUT (4), A) zapisuje informace ze čtvrtého a pátého bitu datové sběrnice, přičemž aktivní jsou v logické jedničce. Pomocí čtvrtého bitu se ovládá připojení příslušné stránky ramdisku a pomocí pátého bitu se ovládá mapování nulté stránky paměti ramdisku do prvních třiceti dvou kB paměťového prostoru, pokud se používá část paměťové kapacity ramdisku jako mapovaná systémová paměť (viz ad 2)). Obvod IO12 je použit jako čtyřbitový registr adresy stránky ramdisku. Jeho výstupy jsou hradlovány signálem RAMDISK, čímž je zabezpečen výběr nulté stránky, pokud není signál RAMDISK aktivní a je komunikováno s pamětí ramdisku.

ad 2) V okamžiku, kdy je požadována komunikace s nultou stránkou paměti ramdisku jako se systémovou pamětí, je aktivní signál EXPRAM (výstup druhé poloviny IO13). Ten aktivuje signál SP1N, pokud je požadován přístup do spodní poloviny systémové paměti [v rovnici (1) je to vyjádřeno druhým sčítancem]. Výstupy registru stránky jsou blokovány pouze při aktivním signálu RAMDISK. V případě, že jsou aktivovány oba signály, je obsazení paměti následující:

00000H až 03FFFFH — spodní polovina nulté stránky ramdisku jako systémová paměť,
04000H až 0BFFFFH — aktuální adresovaná stránka ramdisku,
0C000H až 0FFFFH — vnitřní systémová paměť mikropočítače.



Obr. 1. Řídící logika

Je vidět, že aktivace ramdisku má prioritu, což je nutné kvůli jednoduché komunikaci s ramdiskem. Připínání stránky ramdisku doprostřed adresového prostoru mikropočítače bylo zvoleno proto, aby se zachovala dostupnost počátku paměti, kde jsou umístěny odskoky a pracovní prostor operačního systému, a konce paměti, kde je umístěn systém.

ad 3) Pokud si vytvoříme BIOS tak, že nebude třeba spolupráce s pamětí ROM Spectra, nebo tak, že v něm bude obsažen zaváděč, který do paměti zavade obsah paměti ROM Spectra, můžeme používat zavádění systému z diskety zaváděčem v paměti EPROM. Zavádění systému je inicializováno připojením signálu SBOOT na úroveň log. 0. Tímto signálem se nastaví klopný obvod IO15 (obr. 1) a vygeneruje signál RESETN. Do spodní poloviny paměťového prostoru se připojuje paměť EPROM (obr. 2) a v horní polovině paměti mikroprocesor spolupracuje s horní polovinou vnitřní systémové paměti mikropočítače. Pozor! Při využívání této funkce je nutno zajistit ode-

pnutí paměti PROM s páskovým zaváděčem, nejlépe pomocí dvojitého tlačítka, jehož jedna polovina bude nastavovat IO15 (obr. 1) a druhá polovina bude nulovat klopný obvod IO2 (obr. 1 v [1]), například nulováním signálu CSRESN (obvod 3205 bez potíží vydrží na jednom výstupu krátkodobý zkrat) nebo krátkodobým připojením log. 1 na hodinový vstup tohoto obvodu.

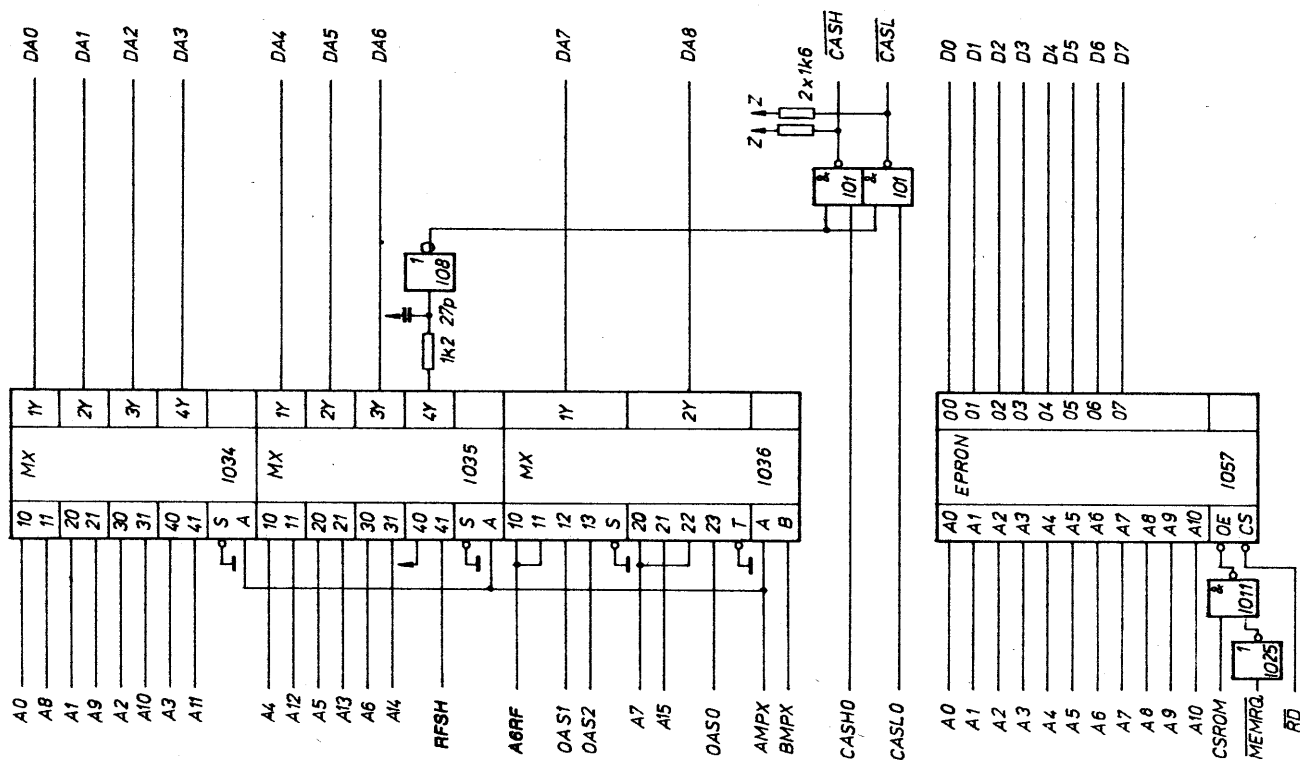
Další obvody na obr. 1 generují signály CASL0 a CASH0 (výběrové signály pro jednotlivé osmice dynamických pamětí) a signály ovládající multiplexery adresy dynamických pamětí. Dvojice klopných obvodů IO14 generuje osmý bit osvězovací adresy dynamických pamětí.

Na obr. 2 je také obvod, který vytváří výběrový signál pro paměť EPROM podle vztahu

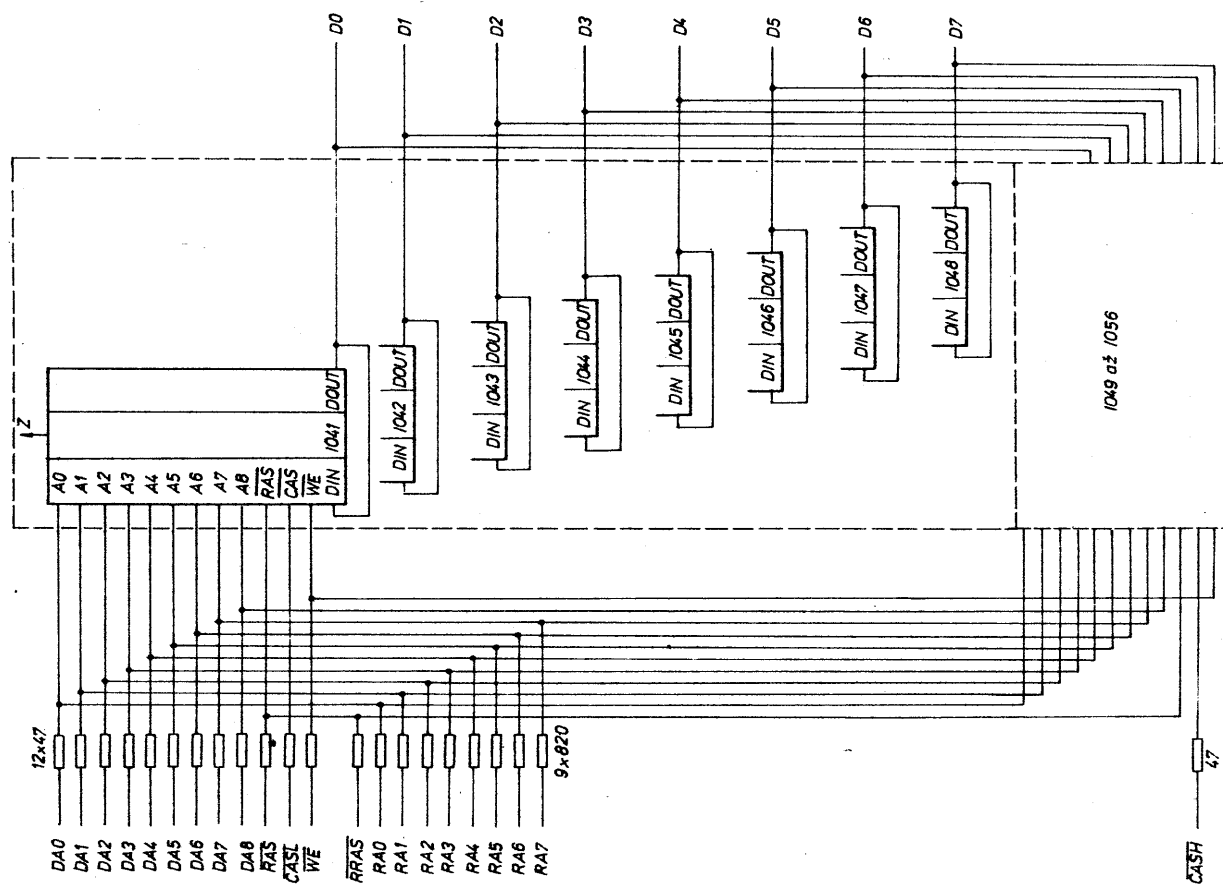
$$CSROM = A15N \cdot BOOT \quad (3)$$

Na obr. 2 je zapojení multiplexerů adresy dynamické paměti RWM, které multiplexují vodiče adresové sběrnice

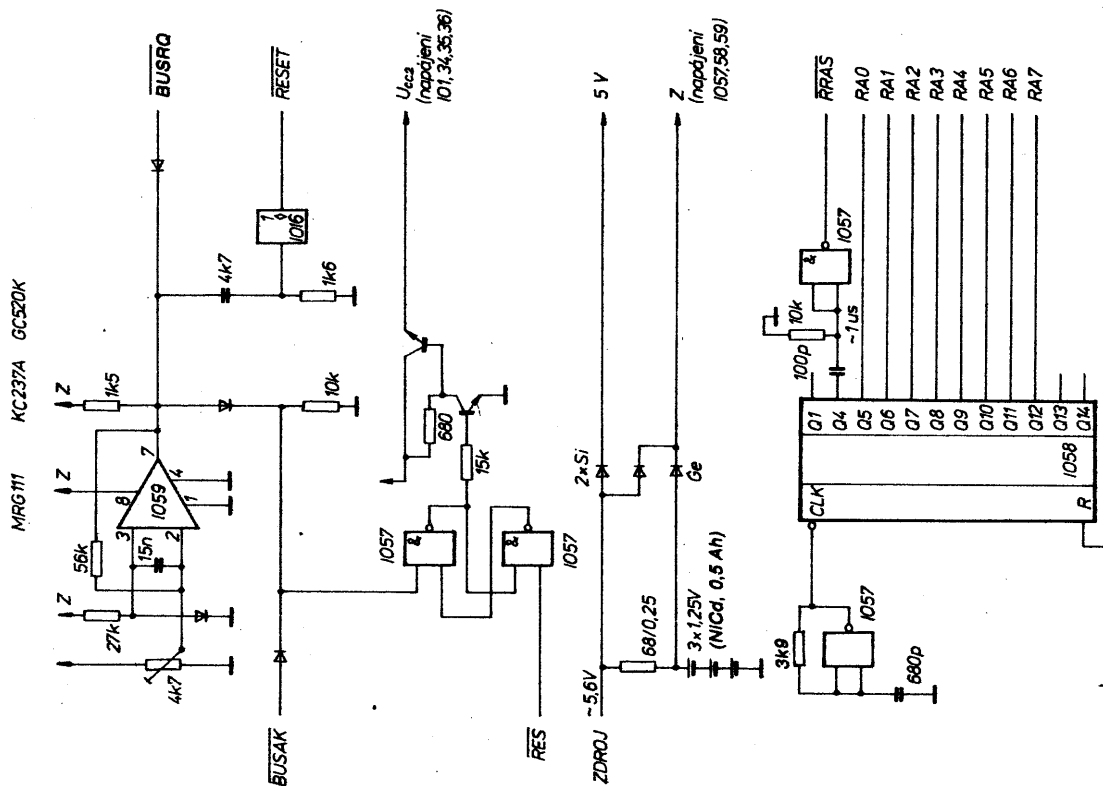
mikropočítače, hradlované výstupy registru adresy stránky ramdisku a osmý bit osvězovací adresy dynamické paměti RWM. Integrovaný člen RC zpožďuje sestupné hrany signálu CASN a tím zpožďuje otevření výstupních datových budičů dynamických pamětí. Jeho použití bylo vynuceno relativně pomalou reakcí obvodu 3212 (IO36 na obr. 3 v [1]) na konec výběrového signálu. Docházelo ke kolizi na datové sběrnici mikropočítače a tím i k havárii celého systému (programové). Dále je na obr. 2 paměť EPROM s disketovým zaváděčem programem. Je naznačeno použití paměti typu 2716 s kapacitou 2 kB, ale je možno použít beze změny zapojení paměť s kapacitou až 32 kB (tj. buď typ 27256 nebo odpovídající kapacitu sestavit z pamětí o kapacitě menší). V tomto případě by bylo možno celý systém včetně paměti ROM Spectra „natahovat“ blokovými přenosy z této paměti EPROM.



Obr. 2. Multiplexery DRAM, EPROM 2 kB



Obr. 3. Zapojení paměti DRAM



Obr. 4. Zálohování paměti

Na obr. 3 je zapojení dynamických pamětí RWM (celkem dvě osmice obvodů 41256 nebo ekvivalentních), dávající celkovou kapacitu 512 kB. Jsou zde také rezistory, které oddělují výstupy osvězovacího čítače (820R) a multiplexerů adresy dynamické paměti RWM (47R). Tuto paměť lze sestavit i z obvodů s menší pamětí, popř. jen z jedné osmice obvodů, je však nutno patřičně přepočítat konstanty ramdisku v tabulce disku A v BIOSu a případně upravit i adresaci pamětí na vstupech multiplexerů adres dynamické paměti.

Na obr. 4 jsou obvody logiky zálohování obsahu dynamické paměti RWM. Základ je tvořen integrovanými obvody — komparátorem napětí IO59, čítačem IO58 (v provedení CMOS) a čtveřicí hradel NAND (také v provedení CMOS).

Pokud je napájecí napětí nad mezí, kdy začíná překlápět komparátor IO59, jsou na adresové vstupy dynamických pamětí přiváděny výstupy multiplexerů IO34—36. V okamžiku dosažení komparační úrovně je procesor požádán o přidělení sběrnic (aktivaci signálu BUSRQ) a v okamžiku potvrzení žádosti (signálem BUSAK) je přeplopen klopný obvod z hradel NAND IO57 a odepnuto napájení IO1, 34, 35 a 36. Tímto postupem se spolehlivě zabrání náhodnému přepsání obsahu dynamických pamětí během nabíhání a vypínání zdroje. Integrované obvody TTL mají bez přítomnosti napájecího napětí velmi velký výstupní odpor, takže se chovají jako by byly ve stavu vysoké impedance („třetí stav“). Experimentálně bylo zjištěno, že toto řešení je spolehlivější, než použití obvodů s třístavovými výstupy, které se při vypínání napájecího napětí mohou mžikově aktivovat a způsobit nežádoucí změnu obsahu některých paměťových buněk.

V průběhu zálohovacího cyklu jsou napájeny pouze obvody pamětí RWM a obvody IO57 až 59. Z jednoho hradla IO57 je vytvořen multivibrátor, který má periodu při napájecím napětí 2,7 V asi 1 μ s. Výstupy čítače IO58 jsou nyní připojeny na adresové vstupy dynamických pamětí, signál RASN je zkrácen asi na 0,5 až 1 μ s při napětí 2,7 V. Nové typy pamětí mají totiž podstatně menší odběr, pokud nejsou aktivovány. Obvody zálohování byly vyzkoušeny s několika různými typy dynamických pamětí od různých výrobců — odběr z náhradního zdroje byl mezi 20 a 45 mA při napětí na pouzdrů pamětí 3,0 V. Při tomto napětí nebyl zjištěn výpadek uchovávané informace ani v jednom případě, každá ze zkoušených osmic pamětí byla takto provozována po dobu dvou hodin.

1.2 Obvody připojení disketové mechaniky

Obvody připojení disketové mechaniky jsou znázorněny na obr. 5 až 7. Pro ovládání disketové mechaniky jsou nezbytně nutné obvody na obr. 5 a 6, obvody na obr. 7 jsou obvody kanálu DMA, které jsou vhodné při vyšších přenosových rychlostech pro záznam na disketu (například mechaniky 5 1/4" s kapacitou 1,2 MB — záznam HD-High Density). Při použití mechanik s poloviční hustotou záznamu, označovaných u formátu IBM DD (Double Density), není řadič kanálu DMA nutný, přenos dat je možno provádět programově.

Na obr. 5 jsou obvody hodinové základny řadiče floppy diskové mechaniky, separátoru dat a prekompenzace zápisu na disketu. Tyto obvody jsou navrženy tak, aby data na disketě byla záznamově kompatibilní se záznamem formátu IBM System 34 Double Density

(MFM), používaným v počítačích řady PC-XT.

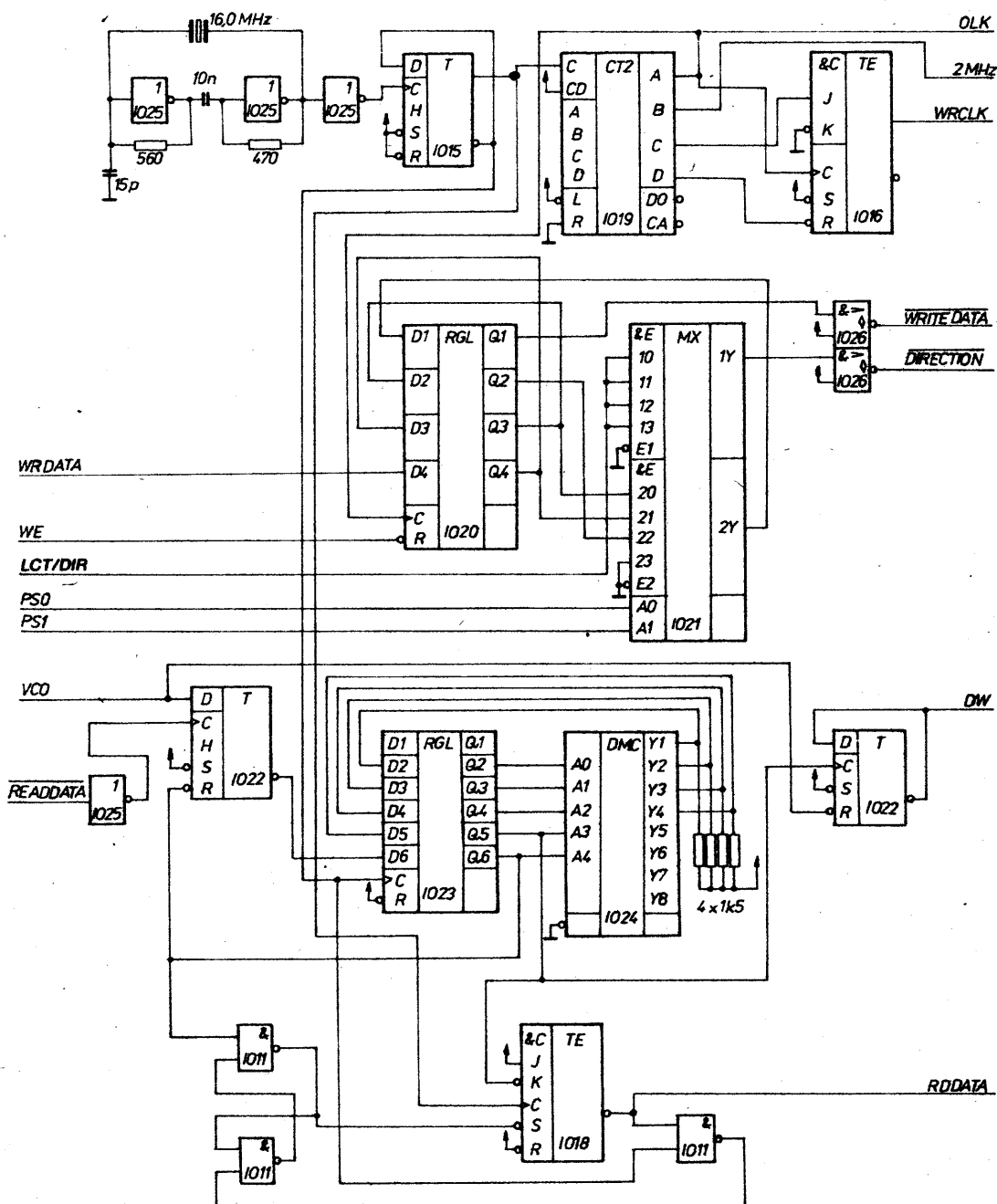
Prvním článkem hodinové základny je krystalem řízený multivibrátor z inverterů TTL, kmitající na kmitočtu 16,000 MHz. Klopným obvodem IO15, šestnáctkovým čítačem IO19 a klopným obvodem IO18 jsou vytvářeny hodinové signály pro řízení číslicového fázového závěsu, řadiče FD a řadiče DMA.

Obvody prekompenzace zápisu dat jsou tvořeny čtyřbitovým registrem IO20 a polovinou dvojitého čtyřkanálového multiplexeru IO21. Mají za úkol zvýraznit odstup mezer a jedniček nahrávaných dat pro snížení vlivu kolísání rychlosti otáčení diskety v disketové mechanice.

Obvody separátoru dat mají za úkol oddělit data od hodin v signálu, jdoucím z diskové mechaniky. Protože záznam na disketu je jednostopý, je nutno použít speciální kódování, při kterém je možno zaznamenat do jednoho přenosového kanálu data i hodiny zároveň. Vzhledem ke kolísání otáček disketové mechaniky totiž nelze zaznamenávat bloky sériových dat bez zakódování synchronizačního hodinového kmitočtu. Obvody separátoru dat se skládají z řízeného číslicového fázového závěsu (IO22, 23, 24) a vlastního separátoru dat, který je tvořen klopným obvodem IO18, hradly IO11 a klopným obvodem IO22. Fázový závěs je řízen signálem VC0 z řadiče, kterým se povoluje zasynchronizování fázového závěsu. Ten se díky obsahu paměti PROM IO24 zasynchronizuje již při příchodu druhého impulsu a je v tomto stavu udržován, pokud je signál VC0 aktivní. Výstupem jsou dva signály

| adresa | data |
|--------|------|
| 00 | 01 |
| 01 | 01 |
| 02 | 02 |
| 03 | 02 |
| 04 | 03 |
| 05 | 03 |
| 06 | 04 |
| 07 | 04 |
| 08 | 0D |
| 09 | 0D |
| 0A | 0E |
| 0B | 0E |
| 0C | 0F |
| 0D | 0F |
| 0E | 00 |
| 0F | 00 |
| 10 | 01 |
| 11 | 02 |
| 12 | 03 |
| 13 | 04 |
| 14 | 05 |
| 15 | 06 |
| 16 | 07 |
| 17 | 08 |
| 18 | 09 |
| 19 | 0A |
| 1A | 0B |
| 1B | 0C |
| 1C | 0D |
| 1D | 0E |
| 1E | 0F |
| 1F | 00 |

Obsah PROM
IO24 (MH74188)



Obr. 5. Hodiny, separátor a prekompenzace zápisu

— DW (data window — datové okno) a RDDATA (read data — čtená data), které jsou vedeny ke zpracování do řadiče typu I8272.

Na obr. 6 je schéma propojení řadiče typu I8272 s ostatními obvody, pomocná hradla řadiče a registr ovládání disketových mechanik. Je vhodné připomenout, že řadič typu I8272 je v RVHP vyráběn v BLR (označení CM 609) a v NDR (U8272D). Dále se na trhu objevuje pod označením μ PD765, např. od firmy NEC. Bližší informace o tomto obvodu najdou zájemci v [4].

Registr ovládání disketových mechanik je tvořen čtyřbitovým registrem IO40. Využity jsou tři bity, dva pro výběr příslušné mechaniky a jeden pro ovládání motoru. Mechaniky musí mít

nastaveny vnitřní propojky tak, aby se zapnutí motoru ovládalo součinem signálů spuštění motoru a výběru mechaniky. Přitažení hlavičky se ovládá součinem signálů výběru mechaniky a vnitřní připravenosti mechaniky. V BIOSu není využíván signál READY mechaniky, protože ne všechny mechaniky jej poskytují, ale počítá se s maximální možnou dobou, za kterou je mechanika od aktivace připravena (u použité mechaniky FD1053 firmy NEC to bylo 0,8 s).

Na obr. 7 je schéma zapojení řadiče kanálu přímého přístupu do paměti (DMA) (není nutný při použití zápisového hodinového kmitočtu 500 kHz). Lze jej ale zapojit a v patci řadiče DMA (8257), pokud není kanál přímého přístupu do paměti využíván, propojit následující vývody na příslušné logické úrovně:

$\overline{\text{DRQ2}}$ — log. 0

TC — log. 0
DACK2N — log. 1
HRQ — log. 0
AEN — log. 0

Obvody řadiče přímého přístupu do paměti jsou tvořeny vlastním řadičem (IO33) a posuvným čtyřbitovým registrem IO30. Ten zajišťuje vždy na jeden požadavek o přenos dat kanálem DMA trvání signálu DRQ2 po dobu čtyř period hodinového signálu. To zabezpečuje přenesení právě jednoho bajtu řadičem. Registr horní poloviny adresy IO31 zajišťuje přítomnost horních osmi adresových bitů na adresové sběrnici, protože horní polovina adresové sběrnice je multiplexována s datovou. Hradla AND obvodu IO29 spolu s oddělovačem IO32 zajišťují jednoduché a v tomto případě postačující překódování řídicí sběrnice mikroprocesoru typu 8080 na řídicí sběrnici Z80 a zpět.



Použité součástky byly z řady 74LS... Co se týče praktického provedení, platí zásady uvedené v [1]. Propojovací kabel musí být co nejkratší, pokud možno typu „twist“ nebo alespoň mezi každými dvěma živými vodiči musí být tažena zem. Musíme si totiž uvědomit, že vlastně prodlužujeme sběrnici procesoru.

2.1. Seznam použitých int. obvodů

27

3. POPIS PROGRAMOVÉ ČÁSTI

Úvodem této části článku je vhodné připomenout základní zásady organizace souborů na disku v operačním systému CP/M. Logická zařízení a části souborů jsou ta, na která se obrací modul BDOS, zatímco modul BIOS zajišťuje propojení požadavků modulu BDOS dále na konkrétní fyzická zařízení (disketové mechaniky) a fyzické soubory na nich. Z tohoto je zřejmé, že při použití vhodných překódovacích algoritmů v modulu BIOS je možná jiná struktura dat na fyzických discích, než kterou předpokládá BDOS. Je dokonce možné mít na každé z šestnácti možných disketových mechanik jiný formát záznamu dat. Operační systém CP/M počítá s následující logickou strukturou záznamu dat na disketu:

věta — záznam dlouhý 128 bajtů,

alokační blok — počet vět (konstanta BLS viz dále), které jsou alokovány (vyhrazeny) pomocí jednoho bitu v bitové alokační mapě disku. Tato konstanta je vždy mocninou dvou, minimálně je rovna osmi.

Fyzická organizace dat na disku je následující:

sektor — minimální záznam, který lze na disk umístit,

stopa — datový objem dosažitelný bez pohybu hlavičky (tato definice má význam pouze pro disketovou mechaniku, avšak i v případě ramdisku se tento pojem používá, i když zde ztrácí svou fyzikální podstatu).

Pokud se nepoužívá klasicky zformátovaná 8" disketa, je ve velké většině případů fyzický sektor větší než věta, je jejím celistvým násobkem. Obdobně alokační blok bývá celistvým násobkem fyzických sektorů. Pro překódování se používá tzv. blokující — deblokuující algoritmus. Kostra použitého algoritmu je převzata z [2].

Výpis programu BIOS (Basic Input-Output System) je připojen. Předpokládá se následující umístění systémových modulů v paměti:

CCP — od adresy 0D600H
BDOS — od adresy 0DE00H
BIOS — od adresy 0EC00H

Ramdisk má kapacitu 480 kB (32 kB je nutná stránka, o níž již byla řeč v předešlé části článku), je formátován po stopách o velikosti 4 kB, první dvě stopy jsou vyhrazeny pro systém, skutečná kapacita ramdisku je tedy 427 kB. S touto kapacitou je také nutno počítat při tvoření konstant v tabulce DPB pro ramdisk — velikost alokačního bloku je zvolena 2 kB (tj. při dané kapacitě nejmenší možný). Pokud se aplikuje jiný ramdisk (o jiné kapacitě), je nutno přepočítat konstanty tabulky DPB (Disk Parameter Block). Totéž platí i pro použití jinak formátované diskety. Jednotlivé položky tabulky DPB mají následující význam:

SPT — (Sectors Per Track) počet sektorů na stopu, přičemž sektorem se

myslí vždy 128 bajtů. Pokud je disketa oboustranná, tak se většinou uvažují jako stopa oba povrchy, tzn. dvojnásobná kapacita.

BSH — (Block SHift) posun bloku. Je to dvojkový logaritmus počtu vět v alokačním bloku.

BLM — (Block Mask) maska bloku. Počet vět v alokačním bloku minus jedna. Je to vlastně adresa poslední věty v alokačním bloku.

EXM — (EXtend Mask) maska rozšíření. Počet logických rozšíření (po 16 kB), adresovaných jednou položkou adresáře minus jedna. Je dána takto:

| BLS | DSM<256 | DSM>=256 |
|-------|---------|----------|
| 1 kB | 0 | — |
| 2 kB | 1 | 0 |
| 4 kB | 3 | 1 |
| 8 kB | 7 | 3 |
| 16 kB | 15 | 7 |

DSM — (Disk Size Max) maximální velikost disku, to je celková kapacita diskové jednotky v alokačních blocích minus jedna (adresa posledního alokačního bloku).

DRM — (DiRectory Max) — maximální velikost adresáře, počet položek adresáře minus jedna. Adresář musí zabírat celistvý počet alokačních bloků. Platí:

$$DRM = (X * (BLM + 1) * 4) - 1$$

kde **X** je počet alokačních bloků adresáře.

ALO, AL1 — (ALlocated 0,1) alokováno 0,1. Jedná se o první dva bajty alokační mapy disku. Jednička v 7. bitu 0. bajtu rezervuje 0. blok atd.

CKS — (Check Sum) kontrolní součet. Počet sektorů adresáře testovaných při zápisu na pracovní médium.

OFF — (OFFset) posunutí. Jedná se o počet stop rezervovaných na počátku disku. Většinou jsou tyto stopy používány pro uložení systému.

Podrobný popis výpisu programu BIOS neuvádím, funkce jednotlivých částí programu je objasněna komentáři přímo ve výpisu zdrojového textu.

Další částí programového vybavení je program výstupu na obrazovku v módu 64 znaků na řádek. Tento program je uveden také ve výpisu zdrojového textu v assembleru, BIOS předpokládá jeho umístění nad videopamětí od adresy 06800H v „zakryté“ části paměti. Proti jiným znakovým výstupům na obrazovku je program doplněn částí umožňující zpracovávání řídících znaků podle ASCII. Před jeho voláním se musí zrušit mapování paměti a po provedení výstupu opět obnovit. Přerušení není nutno zakazovat, je používán přerušovací mód 2 a obsluha se provádí v horní polovině paměti. To však platí pouze pokud máme zaručeno, že ukazatel zásobníku míří do horní poloviny paměti, což není vždy splněno. Z toho důvodu je nejjednodušším řešením přerušení zakázat. Tabulka generátorů znaků je použita z programu D-Writer 666. ZO Svazarmu, je však možno použít každý generátor s českými znaky, který má znaky umístěny v horní polovině bajtu a pro jeden znak potřebuje osm bajtů. Tabulku generátoru znaků získáme z programu D-Writer např. po příkazu EXT.-B příkazem SAVE „genchrsc“ CODE 35708,1064 (nahrajeme na pásku).

2.1 Vytvoření a oživení systému

Pro první pokusy předpokládáme, že máme na pásce nahrané následující programy ve formátu ZX Spectrum:

1. Program znakového výstupu na obrazovku (nahraváme od 06800H).
2. Generátor znaků (nahraváme od 07000H).
3. Modul CCP (nahraváme od 0D600H).
4. Modul BDOS (nahraváme od 0DE00H).
5. Modul BIOS (nahraváme od 0EC00H).

Před jejich nahráváním zakážeme přístup BASICu do oblasti systému např. příkazem CLEAR 24000. Po nahrání všech těchto souborů můžeme teoreticky začít pracovat pod operačním systémem CP/M. Po spuštění systému příkazem RANDOMIZE USR 65408 se smaže obrazovka a vypíše se otázník. Pokud odpovíme y, proběhne formátování ramdisku, jinak není ramdisk formátován (při startu po výpadku napájecího napětí), je vypsáno úvodní hlášení BIOSu a systém se ohlásí např. 0A>. Nyní však potřebujeme disketovou mechaniku a nahanou disketu se základním programovým vybavením. Pokud jsme se nerozhodli pro stavbu řadiče disketové mechaniky (ať už z jakýchkoliv důvodů) a máme pouze ramdisk a kazetový magnetofon, je třeba nejprve vytvořit program TAPE.COM, který nám umožní komunikaci s magnetofonem pod systémem CP/M. Tento program je nutno vytvořit pod operačním systémem Spectra, přeložit, nahrát do paměti a pomocí krátkého pomocného programu ve strojovém kódu jej přenést do části paměti od adresy 00100H. Zde můžeme s výhodou využít možnosti připnout si nultou stránku ramdisku doprostřed systémové paměti a program, který má být nahrán od adresy 00100H do ní nahrajeme od adresy 08100H. Při připnutí nulté stránky jako systémové paměti bude již umístěn od adresy 00100H. Prvním příkazem, který musíme zadat po ohlášení se systémem, je SAVE 8 TAPE.COM, čímž nahrajeme tento program na ramdisk. Můžeme využívat (nyní již poměrně velkou), programovou základnu používanou majiteli počítačů SHARP, pracujících pod operačním systémem CP/M s ramdiskem a kazetovým magnetofonem. Prostřednictvím mgf pásky lze lehce vyměňovat programy a data mezi majiteli těchto dvou různých typů počítačů, po připojení 8" disketové mechaniky lze vyměnit programy prakticky se všemi majiteli profesionálně vyráběných počítačů, pracujících pod systémem CP/M.

4. LITERATURA

- [1] Juřík, A.: Postavte si mikropočítač programově kompatibilní se ZX Spectrum. Ročenka Amatérského radia Mikroelektronika 1988, s. 4.
- [2] Richta, K., Zajíc, J.: Operační systém CP/M pro mikropočítače. Knižnice ČSVTS, svazek 14 díl 1, Praha 1986.
- [3] Johnson, A., Osborne, L.: The programmer's CP/M handbook. McGraw-Hill, London 1983.
- [4] INTEL, Santa Clara, USA: Component Data Catalog. Intel Corporation 1981.

| | | | | | | | | | |
|------|------------------------------|-------------|------|----------------------------|-------------|------|------------------------------|-----------|----------------|
| 1415 | PUSH | HL | 1935 | LD | A,C | 2455 | CHKUNA | LD | A,(UNACNT) |
| 1420 | LD | HL,STAT | 1940 | LD | (SEKSCR),A | 2460 | LD | OR | A |
| 1425 | CONCS2 | | 1945 | SRL | A | 2465 | OR | JR | Z,ALLOC |
| 1430 | CALL | CI | 1950 | SRL | A | 2470 | JR | DEC | A |
| 1435 | CP | "Q" | 1955 | LD | (SEKSEC),A | 2475 | :pokr v neal. zapisu | LD | (UNACNT),A |
| 1440 | JR | Z,CONCS | 1960 | RET | | 2480 | LD | LD | A,(SEKDSK) |
| 1445 | CP | "Q" | 1965 | | | 2485 | LD | LD | HL,UNADSK |
| 1450 | JR | Z,CONCS | 1970 | :preklad cisla sektoru | | 2490 | LD | CP | (HL) |
| 1455 | RES | 0,(HL) | 1975 | :v BC pomoci prekladove | | 2500 | JR | NZ,ALLOC | |
| 1460 | CONCS0 | | 1980 | :tabulky v DE | | 2505 | LD | LD | HL,(UNATRK) |
| 1465 | POP | HL | 1985 | SECTRA | | 2510 | LD | CALL | STCMP |
| 1470 | RET | | 1990 | LD | H,B | 2515 | JR | NZ,ALLOC | |
| 1475 | CONCS | | 1995 | LD | L,C | 2520 | :stejne disky,stejne stopy? | LD | HL,(UNATRK) |
| 1480 | BIT | 0,(HL) | 2000 | LD | A,(SEKDSK) | 2525 | LD | CALL | STCMP |
| 1485 | JR | Z,CONCS1 | 2005 | AND | A | 2530 | JR | NZ,ALLOC | |
| 1490 | RES | 0,(HL) | 2010 | RET | Z | 2535 | :stejny sektor? | LD | A,(SEKSCR) |
| 1495 | JR | CONCS0 | 2015 | EX | DE,HL | 2540 | LD | LD | HL,UNASEC |
| 1500 | CONCS1 | | 2020 | ADD | HL,BC | 2545 | CP | (HL) | |
| 1505 | SET | 0,(HL) | 2025 | LD | L,(HL) | 2550 | JR | NZ,ALLOC | |
| 1510 | SET | 1,(HL) | 2030 | LD | H,0 | 2555 | :priprava adresy pro | | |
| 1515 | JR | CONCS2 | 2035 | RET | | 2560 | :pristi zapis | INC | (HL) |
| 1520 | | | 2040 | | | 2565 | LD | LD | A,(HL) |
| 1525 | :vystup znaku z C na konzolu | | 2045 | :nastaveni adresy DMA na | | 2570 | CP | CPMSPT | |
| 1530 | CONOUT | | 2050 | :adresu z BC | | 2575 | JR | C,NOOVF | |
| 1535 | DI | | 2055 | SETDMA | | 2580 | :dalsi stopa | LD | (HL),0 |
| 1540 | XOR | A | 2060 | LD | (DMAAD),BC | 2585 | LD | LD | HL,(UNATRK) |
| 1545 | OUT | (4),A | 2065 | RET | | 2590 | LD | LD | HL,(UNATRK),HL |
| 1550 | LD | (STACK),SP | 2070 | | | 2600 | :shoda neni treba predecitat | XOR | A |
| 1555 | LD | SP,#6FFF | 2075 | :provedeni operace cteni | | 2605 | NOOVF | LD | (RSFLAG),A |
| 1560 | CALL | COOUT | 2080 | :sektoru z disku | | 2610 | XOR | JR | RWOPER |
| 1565 | LD | SP,(STACK) | 2085 | READ | | 2615 | :zapis do alokovaneho, | LD | A |
| 1570 | LD | A,#20 | 2090 | LD | A,(SEKDSK) | 2620 | :je treba predecist | LD | (UNACNT),A |
| 1575 | OUT | (4),A | 2095 | AND | A | 2625 | LD | INC | A |
| 1580 | EI | | 2100 | JR | NZ,READF | 2630 | LD | LD | (RSFLAG),A |
| 1585 | RET | | 2105 | READR | | 2635 | RWOPER | XOR | A |
| 1590 | | | 2110 | CALL | DMAPOS | 2640 | XOR | LD | (ERFLAG),A |
| 1595 | :tisk znaku z C registru | | 2115 | JR | Z,READR0 | 2645 | LD | LD | A,(SEKSEC) |
| 1600 | LIST | | 2120 | CALL | TRANSR | 2650 | LD | LD | (SEKHST),A |
| 1605 | :vlozit podprogram vystupu | | 2125 | LD | DE,DMABUF | 2655 | :je aktivni vyrovnavaci | LD | HL,HSTACT |
| 1610 | :znaku z C registru na | | 2130 | CALL | READSR | 2660 | :pamet? | LD | A,(HL) |
| 1615 | :tiskarnu | | 2135 | LD | HL,DMABUF | 2665 | LD | LD | (HL),1 |
| 1620 | RET | | 2140 | LD | DE,(DMAAD) | 2670 | OR | A | |
| 1625 | | | 2145 | LD | BC,SEKSIZ | 2675 | JR | Z,FILHST | |
| 1630 | :stav tiskarny | | 2150 | LDIR | | 2680 | :ano. Obsahuje pozadovany | LD | A,(SEKDSK) |
| 1635 | LISTST | | 2155 | XOR | A | 2685 | :sektor? | LD | A,(SEKDSK) |
| 1640 | :vlozit podprogram testu | | 2160 | RET | | 2690 | LD | LD | HL,HSTDSK |
| 1645 | :pripravenosti tiskarny | | 2165 | READR0 | | 2695 | CP | (HL) | |
| 1650 | :prijmout znak | | 2170 | CALL | TRANSR | 2700 | JR | NZ,NOMATC | |
| 1655 | LD | A,-1 | 2175 | LD | DE,(DMAAD) | 2705 | :stejny disk. Stejna stopa? | LD | HL,(HSTTRK) |
| 1660 | RET | | 2180 | CALL | READSR | 2710 | CALL | STCMP | |
| 1665 | | | 2185 | XOR | A | 2715 | JR | NZ,NOMATC | |
| 1670 | :derovani znaku z C | | 2190 | RET | | 2720 | :stejny disk i stopa. | LD | A,(SEKHST) |
| 1675 | PUNCH | | 2195 | READF | | 2725 | :Stejny sektor? | LD | A,(SEKHST) |
| 1680 | :derovac nepripojen | | 2200 | XOR | A | 2730 | LD | LD | HL,HSTSEC |
| 1685 | RET | | 2205 | LD | (UNACNT),A | 2735 | CP | (HL) | |
| 1690 | | | 2210 | INC | A | 2740 | JR | Z,MATCH | |
| 1695 | :cteni znaku ze snimace | | 2215 | LD | (READOP),A | 2745 | :pozadovany jiny sektor nez | LD | A,(HSTWRT) |
| 1700 | :do A | | 2220 | LD | (RSFLAG),A | 2750 | :ten ve vyrovnavaci pameti | LD | A,(SEKDSK) |
| 1705 | READER | | 2225 | LD | A,WRUAL | 2755 | NOMATC | LD | HL,HSTDSK |
| 1710 | :snimac nepripojen, cte se | | 2230 | LD | (WRTYPE),A | 2760 | :napln vyrovnavaci pamet | LD | A,(HSTWRT) |
| 1715 | :vzdy konec zaznamu (CTRL Z) | | 2235 | JP | RWOPER | 2765 | FILHST | LD | A,(SEKDSK) |
| 1720 | LD | A,#1A | 2240 | | | 2770 | LD | LD | (HSTDSK),A |
| 1725 | RET | | 2245 | :provedeni operace zapisu | | 2775 | LD | LD | HL,(SEKTRK),HL |
| 1730 | | | 2250 | :sektoru na disk | | 2780 | LD | LD | A,(SEKHST) |
| 1735 | :podprogramy obsluhy disku | | 2255 | WRITE | | 2785 | LD | LD | HL,HSTSEC |
| 1740 | | | 2260 | LD | A,(SEKDSK) | 2790 | CP | (HL) | |
| 1745 | :nastav nultou stopu | | 2265 | AND | A | 2795 | JR | Z,MATCH | |
| 1750 | :aktualniho disku | | 2270 | JR | NZ,WRITEF | 2800 | :pozadovany jiny sektor nez | LD | A,(HSTWRT) |
| 1755 | HOME | | 2275 | WRITER | | 2805 | :ten ve vyrovnavaci pameti | LD | A,(SEKHST) |
| 1760 | LD | A,(HSTWRT) | 2280 | CALL | DMAPOS | 2810 | NOMATC | LD | HL,HSTSEC |
| 1765 | OR | A | 2285 | JR | Z,WRITR0 | 2815 | LD | LD | HL,(HSTTRK) |
| 1770 | JR | NZ,HOMED | 2290 | LD | DE,DMABUF | 2820 | LD | LD | A,(SEKHST) |
| 1775 | LD | (HSTACT),A | 2295 | LD | HL,(DMAAD) | 2825 | LD | LD | (HSTSEC),A |
| 1780 | HOMED | | 2300 | LD | BC,SEKSIZ | 2830 | OR | A | |
| 1785 | XOR | A | 2305 | LDIR | | 2835 | CALL | NZ,WRHST | |
| 1790 | LD | (SEKTRK),A | 2310 | CALL | TRANSR | 2840 | :napln vyrovnavaci pamet | LD | A,(SEKDSK) |
| 1795 | RET | | 2315 | LD | DE,DMABUF | 2845 | FILHST | LD | (HSTDSK),A |
| 1800 | | | 2320 | EX | DE,HL | 2850 | LD | LD | HL,(SEKTRK) |
| 1805 | :vyber diskove mechaniky | | 2325 | CALL | READSR | 2855 | LD | LD | (HSTTRK),HL |
| 1810 | :specifikovane C registrem | | 2330 | XOR | A | 2860 | LD | LD | A,(SEKHST) |
| 1815 | SELDSK | | 2335 | RET | | 2865 | LD | LD | (HSTSEC),A |
| 1820 | LD | A,C | 2340 | WRITR0 | | 2870 | LD | LD | A,(RSFLAG) |
| 1825 | CP | Z | 2345 | CALL | TRANSR | 2875 | OR | A | |
| 1830 | LD | HL,0 | 2350 | LD | DE,(DMAAD) | 2880 | CALL | NZ,RDHST | |
| 1835 | RET | NC | 2355 | EX | DE,HL | 2885 | XOR | A | |
| 1840 | LD | (SEKDSK),A | 2360 | CALL | READSR | 2890 | LD | LD | (HSTWRT),A |
| 1845 | LD | L,A | 2365 | XOR | A | 2895 | LD | LD | A,C |
| 1850 | ADD | HL,HL | 2370 | RET | | 2900 | LD | LD | (WRTYPE),A |
| 1855 | ADD | HL,HL | 2375 | WRITEF | | 2905 | LD | LD | A,(SEKSCR) |
| 1860 | ADD | HL,HL | 2380 | XOR | A | 2910 | LD | LD | SECMK |
| 1865 | ADD | HL,HL | 2385 | LD | (READOP),A | 2915 | LD | LD | L,0 |
| 1870 | LD | DE,DPBASE | 2390 | LD | A,C | 2920 | LD | LD | H,A |
| 1875 | ADD | HL,DE | 2395 | LD | (WRTYPE),A | 2925 | LD | LD | H |
| 1880 | RET | | 2400 | CP | WRUAL | 2930 | LD | LD | L |
| 1885 | | | 2405 | JR | NZ,CHKUNA | 2935 | LD | LD | DE,HSTBUF |
| 1890 | :nastaveni stopy dane | | 2410 | LD | A,BLSIZ/128 | 2940 | LD | LD | HL,DE |
| 1895 | :registrem C | | 2415 | LD | (UNACNT),A | 2945 | LD | LD | DE,HL |
| 1900 | SETTRK | | 2420 | LD | A,(SEKDSK) | 2950 | LD | LD | HL,(DMAAD) |
| 1905 | LD | (SEKTRK),BC | 2425 | LD | (UNADSK),A | 2955 | LD | LD | BC,128 |
| 1910 | RET | | 2430 | LD | HL,(SEKTRK) | | | | |
| 1915 | | | 2435 | LD | (UNATRK),HL | | | | |
| 1920 | :nastaveni sektoru | | 2440 | LD | A,(SEKSCR) | | | | |
| 1925 | :daneho registrem C | | 2445 | LD | (UNASEC),A | | | | |
| 1930 | SETSEC | | 2450 | :test pokr. v neal. zapisu | | | | | |

```

2975 LD A,(READOP)
2980 OR A
2985 JR NZ,RWMOVE
2990 ;zapis, oznac a obrat smer
2995 LD A,1
3000 LD (HSTWRT),A
3005 EX DE,HL
3010 ;presun 128 byte
3015 ;z (HL) do (DE)
3020 RMWMOVE
3025 EX DE,HL
3030 LDIR
3035 ;data byla presunuta z/do
3040 ;HSTBUF
3045 LD A,(WRTYPE)
3050 CP WRDIR
3055 LD A,(ERFLAG)
3060 RET NZ
3065 ;zapis do adresare je treba
3070 ;provest ihned
3075 OR A
3080 RET NZ
3085 XOR A
3090 LD (HSTWRT),A
3095 CALL WRHST
3100 LD A,(ERFLAG)
3105 RET
3110
3115 ;prevod logickeho cisla
3120 ;sektoru na fyzicke
3125 ;pro ramdisk
3130 ;vstup: (SEKSEC),(SEKTRK)
3135 ;vystup: HL - pocatecni
3140 ; adresa sektoru o
3145 ; delce 128 byte
3150 TRANSR
3155 LD A,(SEKTRK)
3160 LD E,0
3165 LD D,A
3170 SRL D
3175 RR E
3180 SRL D
3185 RR E
3190 SRL D
3195 RR E
3200 ;v D fyzicky track,
3205 ;v E nizsi bity adresy
3210 LD A,D
3215 ADD A,RDSKOF
3220 OUT (3),A
3225 SRL E
3230 LD HL,#4000
3235 LD A,E
3240 ADD A,H
3245 LD H,A
3250 LD A,(SEKSCR)
3255 SRL A
3260 RR L
3265 OR H
3270 LD H,A
3275 RET
3280
3285 ;zjisteni nutnosti pouziti
3290 ;vyrovnavaciho bufferu pro
3295 ;operaci s ramdiskem
3300 ;vystup: Z - 0 neni treba
3305 ; 1 je treba
3310 DMAPOS
3315 LD DE,(DMAAD)
3320 LD HL,128
3325 ADD HL,DE
3330 LD A,H
3335 AND #C0
3340 RET Z
3345 LD A,D
3350 AND #C0
3355 CP #C0
3360 RET
3365
3370 ;podprogram cteni/zapisu
3375 ;sektoru z ramdisku
3380 READSR
3385 LD BC,128
3390 LD A,#30
3395 DI
3400 OUT (4),A
3405 LDIR
3410 LD A,#20
3415 OUT (4),A
3420 EI
3425 RET
3430
3435 ;podprogram pro 16 bitove
3440 ;porovnani
3445 ;vstup: HL - (UNATRK) nebo
3450 ; (HSTTRK)
3455 ; (SEKTRK)
3460 ;vystup: Z - 0 stejne
3465 ; 1 ruzne
3470 STCMF
3475 LD DE,(SEKTRK)
3480 LD A,L
3485 CP E
3490 RET NZ

```

```

3495 LD A,H
3500 CP D
3505 RET
3510
3515 ;podprogram povelu FDC
3520 ;vstup: HL - adresa pocatku
3525 ; datoveho bloku
3530 FDCOUT
3535 LD C,FDCDTR
3540 LD B,(HL)
3545 INC HL
3550 FDCOU2
3555 IN A,(FDCMSR)
3560 AND #C0
3565 CP #80
3570 JR NZ,FDCOU2
3575 OUTI
3580 JR NZ,FDCOU2
3585 IN A,(FDCMSR)
3590 RET
3595
3600 ;podprogram vstupu vysledku
3605 ;po provedeni povelu FDC
3610 ;vstup: HL - adresa pocatku
3615 ; ukladani dat
3620 ; B - delka bloku
3625 FDCIN
3630 LD C,FDCDTR
3635 FDCI_0
3640 IN A,(FDCMSR)
3645 AND #C0
3650 CP #C0
3655 JR NZ,FDCI_0
3660 INI
3665 JR NZ,FDCI_0
3670 RET
3675
3680 ;podprogramy fyzickeho
3685 ;zapisu nebo cteni
3690 ;sektoru floppydisku
3695 ;vstup: (HSTDSK) - disk
3700 ; (HSTTRK) - stopa
3705 ; (HSTSEC) - sektor
3710 ; pres WRHST - zapis
3715 ; pres RDHST - cteni
3720 ;zapisuje/cte se HSTSIZ
3725 ;byte z/do HSTBUF
3730 ;vstup: nastaveni (ERFLAG)
3735 WRHST
3740 LD A,#45
3745 LD L,#80
3750 JR FDCNST
3755 RDHST
3760 LD A,#46
3765 LD L,#40
3770 FDCNST
3775 LD (FDCRW+1),A
3780 LD A,L
3785 LD (DMAREG),A
3790 XOR A
3795 LD (ERFLAG),A
3800 LD A,RETRY
3805 LD (PRETRY),A
3810 FDCNS4
3815 CALL WAIT
3820 CALL SELTRK
3825 LD A,(HSTTRK)
3830 LD (FDCRW+3),A
3835 LD A,(HSTSEC)
3840 LD B,A
3845 SUB 9
3850 JR NC,FDCNS0
3855 LD A,B
3860 INC A
3865 LD (FDCRW+5),A
3870 XOR A
3875 JR FDCNS1
3880 FDCNS0
3885 INC A
3890 LD (FDCRW+5),A
3895 LD A,1
3900 FDCNS1
3905 LD (FDCRW+4),A
3910 RLCA
3915 RLCA
3920 LD B,A
3925 LD A,(HSTDSK)
3930 OR B
3935 LD (FDCRW+2),A
3940 LD HL,FDCRW
3945 DI
3950 CALL FDCOUT
3955 LD A,(DMAREG)
3960 BIT 7,A
3965 LD HL,HSTBUF
3970 LD C,FDCDTR
3975 LD B,HSTBLK
3980 CALL WRHS
3985 EI
3990 LD (ERFLAG),A
3995 LD B,7
4000 LD HL,FDCRES
4005 CALL FDCIN
4010 LD A,(FDCRES)

```

```

4015 AND #C0
4020 JR Z,FDCNS2
4025 LD A,1
4030 FDCNS3
4035 LD (ERFLAG),A
4040 LD A,(ERFLAG)
4045 OR A
4050 JR NZ,FDCNS5
4055 LD A,(ERFLAG)
4060 OR A
4065 JR NZ,FDCNS5
4070 RET
4075 FDCNS2
4080 XOR A
4085 JR FDCNS3
4090 FDCNS5
4095 LD A,(PRETRY)
4100 DEC A
4105 LD (PRETRY),A
4110 LD A,-1
4115 RET Z
4120 JR FDCNS4
4125 WRHS
4130 JR NZ,WRHSTW
4135 WRHSTR
4140 LD B,SEKSIZ
4145 WRHSR1
4150 IN A,(FDCMSR)
4155 CP #F0
4160 JR Z,WRHSR2
4165 CP #D0
4170 JR Z,WRHSR9
4175 WRHSR1
4180 WRHSR2
4185 INI
4190 JR NZ,WRHSR1
4195 DEC D
4200 JR NZ,WRHSTR
4205 OUT (TERCNT),A
4210 XOR A
4215 RET
4220 WRHSR9
4225 LD A,-1
4230 RET
4235 WRHSTW
4240 LD B,SEKSIZ
4245 WRHSW1
4250 IN A,(FDCMSR)
4255 CP #B0
4260 JR Z,WRHSW2
4265 CP #D0
4270 JR Z,WRHSR9
4275 WRHSW1
4280 WRHSW2
4285 OUTI
4290 JR NZ,WRHSW1
4295 DEC D
4300 JR NZ,WRHSTW
4305 OUT (TERCNT),A
4310 XOR A
4315 RET
4320
4325 ;podprogram nastaveni DMA
4330 ;kontroleru 8257
4335 ; vstup: L - bity R/W z TC
4340 ; DE - poc. adresa
4345 ; BC - delka bloku
4350 STDMAC
4355 LD BC,HSTSIZ
4360 LD DE,HSTBUF
4365 STDNCO
4370 LD A,#44
4375 OUT (28),A
4380 LD A,C
4385 OUT (25),A
4390 LD A,B
4395 OR L
4400 OUT (25),A
4405 LD A,E
4410 OUT (24),A
4415 LD A,D
4420 OUT (24),A
4425 RET
4430
4435 ;podprogram nastaveni stopy
4440 ;floppy disku z HSTTRK
4445 ;vstup: (ERFLAG),(PCN)
4450 SELTRK
4455 LD A,(HSTTRK)
4460 OR A
4465 JR Z,HOMEF
4470 LD (FDCSC+3),A
4475 LD A,(HSTDSK)
4480 LD (FDCSC+2),A
4485 LD HL,FDCSC
4490 CALL FDCOUT
4495 JR HOMEF0
4500 ;podprogram fyzickeho
4505 ;nastaveni mechaniky na
4510 ;nulovou stopu
4515 HOMEF
4520 CALL WAIT
4525 LD A,(SEKDSK)
4530 LD (FDCRW+2),A

```

```

4535 LD HL,FDCHOM
4540 CALL FDCOUT
4545 HOMEFO IN A,(FDCMSR)
4550 BIT 7,A
4555 JR Z,HOMEFO
4560 LD A,8
4570 OUT (FDCDTR),A
4575 HOMEF1 IN A,(FDCMSR)
4580 BIT 7,A
4590 JR Z,HOMEF1
4595 IN A,(FDCDTR)
4600 BIT 5,A
4605 PUSH AF
4610 HOMEF2 IN A,(FDCMSR)
4615 BIT 7,A
4620 JR Z,HOMEF2
4625 IN A,(FDCDTR)
4630 POP AF
4635 JR Z,HOMEFO
4640 RET
4645
4650
4655 ;podprogram formatovani
4660 ;ramdisku
4665 FORMR DI
4670 LD A,RDSKOF
4680 OUT (3),A
4685 LD A,#30
4690 OUT (4),A
4695 LD HL,#4000
4700 LD DE,#4001
4705 LD BC,#7FFF
4710 LD (HL),#E5
4715 LDIR
4720 LD A,#20
4725 OUT (4),A
4730 RET
4735
4740 ;podprogram vystupu zpravy
4745 ;na konzolu
4750 TEXTCD LD A,(HL)
4755 OR A
4760 RET
4765 Z
4770 INC HL
4775 PUSH HL
4780 LD C,A
4785 CALL CONOUT
4790 HL
4795 JR TEXTCD
4800
4805 ;podprogram spousteni
4810 ;disketove mechaniky
4815 WAIT DI
4820 LD HL,FDCTIM
4825 LD A,(HL)
4830 OR A
4835 JR NZ,WAIT0
4840 LD A,5
4850 OUT (6),A
4855 LD A,FDCTMR*2
4860 LD (HL),A
4865 WAIT0 EI
4870 LD A,(HL)
4875 FDCTMR+1
4880 CP NC,WAIT0
4885 JR LD (HL),FDCTMR
4890 RET
4905
4910
4915 ;podprogram formatovani
4920 ;floppydisku
4925 FORMF CALL HOMEF
4930 LD A,39
4935 FORMFO LD (HSTTRK),A
4940 PUSH AF
4945 CALL SELTRK
4950 CALL FORMTR
4955 POP AF
4960 OR A
4965 RET
4970 Z
4975 DEC A
4980 JR FORMFO
4985
4990
4995 ;formatovani jedne stopy
5000 FORMTR XOR A
5005 LD (HEAD),A
5010 FORMT1 DI
5015 LD A,1
5020 LD (FDCFRM+2),A
5025 FORMT0 LD (SECTOR),A
5030 LD A,(SEKTRK)
5035 LD C,A
5040 CALL FDCOC
5045 LD A,(HEAD)

```

```

5065 LD C,A
5070 CALL FDCOC
5075 LD A,(SECTOR)
5080 LD C,A
5085 CALL FDCOC
5090 LD C,2
5095 CALL FDCOC
5100 LD A,(SECTOR)
5105 INC A
5110 CP 10
5115 JR NZ,FORMT0
5120 LD HL,FDCRES
5125 LD B,7
5130 CALL FDCIN
5135 LD A,(FDCRES)
5140 AND #C0
5145 CP #80
5150 JR Z,FORMTF
5155 LD A,(HEAD)
5160 OR A
5165 EI
5170 RET
5175 INC A
5180 JR FORMT1
5185 FDCOC IN A,(FDCMSR)
5190 AND #C0
5195 CP #80
5200 JR NZ,FDCOC
5210 LD A,C
5215 OUT (FDCDTR),A
5220 RET
5225
5230 ;podprogram hlaseni chyby
5235 ;v prubehu formatovani
5240 FORMTF LD HL,HLASFO
5245 CALL TEXTCD
5250 LD A,(SEKTRK)
5255 CALL CONUM
5260 LD HL,HLASEN
5265 CALL TEXTCD
5270 RET
5275 CONUM PUSH AF
5280 CALL CONUM1
5285 POP AF
5290 RRCA
5295 RRCA
5300 RRCA
5305 RRCA
5310 RRCA
5315 CONUM1 AND #0F
5320 ADD A,6
5325 BIT 4,A
5330 JR NZ,CONUM9
5335 SUB 6
5340 JR CONUM8
5345 INC A
5350 CONUM9 ADD A,#30
5355 LD C,A
5360 CALL CONOUT
5365 RET
5370 HLASFO DEFB "Bios err: "
5375 DEFB "format error"
5380 DEFB 0
5385 HLASEN DEFB CCR,CLF,0
5390
5395 ;podprogram obsluhy
5400 ;preruseni
5405 INTERT DI
5410 PUSH AF
5415 BC
5420 DE
5425 HL
5430 PUSH IY,STAT
5435 CALL KEYB
5440 LD HL,(TIMER)
5445 INC HL
5450 LD (TIMER),HL
5455 LD HL,FDCTIM
5460 LD A,(HL)
5465 OR A
5470 JR Z,INTER0
5475 DEC A
5480 LD (HL),A
5485 OUT (6),A
5490 POP IY
5495 HL
5500 POP DE
5505 POP BC
5510 EI
5515 RET
5520
5525
5530
5535 INTER0 POP IY
5540 HL
5545 POP DE
5550 POP BC
5555 EI
5560 RET
5565
5570
5575
5580

```

```

5585 ;rutina volana pri kazdem
5590 ;preruseni
5595 KEYB LD HL,KS0
5600 KLOOP LD B,(HL)
5605 INC B
5610 JR Z,KCH
5615 INC HL
5620 DEC HL
5625 DEC HL
5630 JR NZ,KCH
5635 LD (HL),-1
5640
5645 KCH LD A,L
5650 LD HL,KS4
5655 CP L
5660 JR NZ,KLOOP
5665 CALL CIN
5670 OR A
5675 RET
5680 LD HL,KS0
5685 CP HL
5690 JR Z,KREP
5695 LD HL,KS4
5700 CP HL
5705 JR Z,KREP
5710 LD B,(HL)
5715 INC B
5720 JR Z,KNEW
5725 LD HL,KS9
5730 CP HL
5735 INC HL
5740 LD A,E
5745 LD (HL),A
5750 KNEW LD E,A
5755 LD (HL),A
5760 INC HL
5765 LD HL,5
5770 INC HL
5775 LD HL,A,(KS9)
5780 LD (HL),A
5785 INC HL
5790 LD A,E
5795 LD (HL),A
5800 KEND LD (KS8),A
5805 LD S,(IY+0)
5810 RET
5815
5820 KREP INC HL
5825 LD HL,5
5830 INC HL
5835 DEC HL
5840 RET
5845 LD NZ,(KSA)
5850 LD (HL),A
5855 LD HL,A,(HL)
5860 LD KEND
5865 ;klavesnice CP/M Spectrum
5870 ;(C) DaJe, 1987
5875 CIN LD BC,#FEFE
5880 LD D,0
5885 KLOOP1 LD A,D
5890 LD E,#1F
5895 CP 0
5900 JR NZ,DAL1
5905 LD E,#1E
5910 DAL1 CP 7
5915 JR NZ,DAL2
5920 LD E,#1D
5925 DAL2 IN A,(C)
5930 CPL
5935 AND E
5940 JR Z,NONE
5945 ;A - priznak rady
5950 ;D - priznak sloupce 0-7
5955 KEY LD C,D
5960 LD H,0
5965 LD L,H
5970 LD B,0
5975 LD D,L
5980 LD E,8
5985 LOOP2 RRCA
5990 JR C,TRANS1
5995 ADD HL,DE
6000 LOOP2
6005 ;HL=8*RADA
6010 ;BC=SLOUPEC
6015 TRANS1 ADD HL,BC
6020 CALL KCAPS
6025 NZ,NOEXTD
6030 CALL SYMBOL
6035 JR NZ,NOEXTD
6040
6045
6050
6055
6060
6065
6070
6075
6080
6085
6090
6095
6100

```

```

6105 LD DE,120
6110 ADD HL,DE
6115 JR NOSYM
6120 NOEXTD
6125 CALL KCAPS
6130 JR NZ,NOCAPS
6135 LD DE,40
6140 ADD HL,DE
6145 NOCAPS
6150 CALL SYMBOL
6155 JR NZ,NOSYM
6160 LD DE,80
6165 ADD HL,DE
6170 NOSYM
6175 LD DE,TABLE
6180 BIT 1,(IY+0)
6185 JR Z,TRANS2
6190 LD DE,TABLE+160
6195 TRANS2
6200 ADD HL,DE
6205 LD A,(HL)
6210 JR EXIT
6215 KCAPS
6220 LD BC,#FEFE
6225 IN A,(C)
6230 BIT 0,A
6235 RET
6240 SYMBOL
6245 LD BC,#7FFE
6250 IN A,(C)
6255 BIT 1,A
6260 RET
6265 NONE
6270 INC D
6275 RLC B
6280 JR C,KLOOP1
6285 ;zadna klapka
6290 XOR A
6295 BIT 0,(IY+0)
6300 JR NZ,EXIT
6305 RES 1,(IY+0)
6310 EXIT
6315 LD C,A
6320 LD B,0
6325 RET
6330
6335 TABLE
6340 ;MALA PISMENA 40 BYTE
6345 DEFB 0
6350 DEFM "a410p"
6355 DEFB 13,32
6360 DEFM "zsw290l"
6365 DEFB 0
6370 DEFM "xds38ikm"
6375 DEFM "cfr47ujn"
6380 DEFM "vgt56yhb"
6385 ;VELKA PISMENA 40 BYTE
6390 DEFB 0
6395 DEFM "AQ10P"
6400 DEFB 13,32
6405 DEFM "ZSW290L"
6410 DEFB 0
6415 DEFM "XDES38IKM"
6420 DEFM "CFR47UJN"
6425 DEFM "VGT56YHB"
6430 ;SHIFTOVANE ZNAKY 40 BYTE
6435 DEFB 0
6440 DEFB 27
6445 DEFB "!"
6450 DEFB "!",13,32
6455 DEFM "!"
6460 DEFM "!"
6465 DEFM "!"
6470 DEFM "!"
6475 ;RIDICI ZNAKY 40 BYTE
6480 DEFB 0,1,17,1
6485 DEFB 0,16,13,32
6490 DEFB 26,19,23,2
6495 DEFB 9,15,12,0
6500 DEFB 24,4,5,3
6505 DEFB 8,9,11,13
6510 DEFB 3,6,18,4
6515 DEFB 7,21,10,14
6520 DEFB 22,7,20,5
6525 DEFB 6,25,8,2
6530 ;MALA CESTINA
6535 DEFB 0,#C1,"q","1"
6540 DEFB 0,"p",13,32
6545 DEFB #DA,#D3,#C5
6550 DEFB 2,"q",#CF
6555 DEFB "1",0,"x",#C4
6560 DEFB #D7,"z",8
6565 DEFB #C9,"k",#m
6570 DEFB #C3,"r",#D2
6575 DEFB 4,"r",#CA
6580 DEFB #D5,#CE,"v"
6585 DEFB 9,"D4",5
6590 DEFB 6,"D9",h
6595 DEFB b
6600 ;VELKA CESTINA
6605 DEFB 0,#E1,"Q","1"
6610 DEFB 0,"P",13,32
6615 DEFB #FA,#F3,#E5
6620 DEFB 2,"9",#EF
6625 DEFB "L",0,"X",#E4
6630 DEFB #F7,"3",8
6635 DEFB #E9,"K",#M
6640 DEFB #E3,"F",#F2
6645 DEFB 4,"7",#EA
6650 DEFB #F5,#EE,"V"
6655 DEFB 6,"#F4",5
6660 DEFB "A",#F9,"H"
6665 DEFB "B"
6670 CI
6675 PUSH BC
6680 PUSH DE
6685 PUSH HL
6690 PUSH IY
6695 LD IY,STAT
6700
6705 WAIT2
6710 CALL INKEY
6715 OR A
6720 JR Z,WAIT2
6725 PRESS
6730 PUSH AF
6735 BEEP
6740 LD C,20
6745 LD D,15
6750 DI
6755 BEEP0
6760 LD B,D
6765 BEEP1
6770 EX (SP),IY
6775 EX (SP),IY
6780 EX (SP),IY
6785 EX (SP),IY
6790 DJNZ BEEP1
6795 LD A,C
6800 AND 1
6805 RLCA
6810 RLCA
6815 RLCA
6820 RLCA
6825 OR 7
6830 OUT (#FE),A
6835 DEC C
6840 JR NZ,BEEP0
6845 POP AF
6850 RES 5,(IY+0)
6855 POP IY
6860 POP HL
6865 POP DE
6870 POP BC
6875 EI
6880 RET
6885 INKEY
6890 PUSH IY
6895 PUSH BC
6900 PUSH DE
6905 PUSH HL
6910 LD IY,STAT
6915 BIT 5,(IY+0)
6920 JR A,(KS8)
6925 LD NZ,FIN
6930 XOR A
6935 FIN
6940 POP HL
6945 POP DE
6950 POP BC
6955 POP IY
6960 RET
6965
6970
6975 ;nasleduje pracovni oblast
6980 ;BIOSu
6985 ORG #F602
6990 BEG DAT EQU 0
6995
7000 DMAAD DEFS 2
7005 DIRBUF DEFS SEKSIZ
7010 ALLOR DEFS 28
7015 ALLOR DEFS 23
7020 CHKOR DEFS 16
7025 CHKOR DEFS 16
7030 DMABUF DEFS SEKSIZ
7035 STACK DEFS 2
7040 TIMER DEFM 0
7045 FDCIM DEFB 0
7050 STAT DEFB 0
7055 KS0 DEFB 0,0,0,0
7060 KS4 DEFB 0,0,0,0
7065 KS8 DEFB 0
7070 KS9 DEFB 20
7075 KSA DEFB 5
7080 PRETRY DEFS 1
7085 DMAREG DEFS 1
7090
7095 ;nasleduje pracovni oblast
7100 ;blokujiciho/deblokujiciho
7105 ;algoritmu pro obsluhu
7110 ;floppy disku 5 1/4"
7115 ;18 sek/trk, 40trks
7120
7125 ;zadana diskova adresa vety
7130 ;SEKDSK - ulozeno na adr. 4
7135 SEKTRK DEFS 2
7140 SEKSEC DEFS 1
7145 SEKSCR DEFS 1
7150 ;diskova adresa fyz. sekt.
7155 HSTDGK DEFS 1
7160 HSTTRK DEFS 2
7165 HSTSEC DEFS 1
7170 ;cislo pozadovaneho fyz.
7175 ;sektoru
7180 SEKHST DEFS 1
7185 ;priznak HSTBUF aktivni
7190 HSTACT DEFS 1
7195 ;priznak odlozeneho zapisu
7200 HSTWRT DEFS 1
7205 ;citac nealokovanych zapisu
7210 UNACNT DEFS 1
7215 ;diskova adresa dalsi vety
7220 ;v nealokovanem bloku
7225 UNADSK DEFS 1
7230 UNATRK DEFS 2
7235 UNASEC DEFS 1
7240 ;kod chyby
7245 ERFLAG DEFS 1
7250 ERFLG0 DEFS 1
7255 ;priznak pozadavku na cteni
7260 RSFLAG DEFS 1
7265 ;priznak cteci operace (=1)
7270 READOP DEFS 1
7275 ;typ zapisove operace
7280 WRTYPE DEFS 1
7285 ;vyrovnavaci pamet
7290 HSTBUF DEFS HSTSIZ
7295
7300 ;nasleduje pracovni oblast
7305 ;obsluhy radice floppy
7310 ;disku typu 18272
7315
7320 ;R/W
7325 FDCRW
7330 DEFB FDCFRM-X-1
7335 DEFB 0,0,0,0,0
7340 DEFB N,EOT,GPL,RW
7345 DEFB DTL
7350 ;format
7355 FDCFRM
7360 DEFB FDCSC-X-1
7365 DEFB #4D,1,N,SC
7370 DEFB GPL,FT,DD
7375 ;set cylinder
7380 FDCSC
7385 DEFB FDCSIS-X-1
7390 DEFB #0F,0,0
7395 ;sense interrupt status
7400 FDCSIS
7405 DEFB FDCCHOM-X-1
7410 DEFB #08
7415 ;home
7420 FDCCHOM
7425 DEFB FDCSPE-X-1
7430 DEFB #07,0
7435 ;specify - nastaveni
7440 ;pocatecnich parametru FDC
7445 FDCSPE
7450 DEFB PCN-X-1
7455 DEFB #03,SRTHTUT
7460 DEFB HLTDND
7465 ;present cylinder number
7470 PCN
7475 DEFS 1
7480 ;aktivni hlava
7485 HEAD
7490 DEFS 1
7495 ;sektor
7500 SECTOR
7505 DEFS 1
7510 ;misto pro ulozeni vysledku
7515 ;operace FDC
7520 FDCRES
7525 DEFS 7
7530
7535 ENDDAT EQU 8
7540 DATSIZ EQU 8-BEGDAT
7545
7550 ORG #FCFC
7555 DI
7560 JP INTERT
7565 ;vstup do systemu po
7570 ;zavedeni z pasky
7575 ORG #FF80
7580 DI
7585 LD A,#20
7590 OUT (4),A
7595 LD A,#F8
7600 LD 1,A
7605 LD HL,INTERT
7610 LD (0FEFF),HL
7615 IM 2
7620 LD SP,#FF7F
7625 LD HL,#F800
7630 LD BC,256
7635 LD (HL),#FC
7640 LD DE,#F801
7645 LDIR
7650 LD C,CFF
7655 CALL CONOUT
7660 LD C,?

```

```

7665 CALL CONOUT
7670 EI
7675 CALL CONIN
7680 CALL CONIN
7685 CP
7690 JR NZ,PRECHO
7695 CALL FORMR
7700 CALL SAVESY
7705 PRECHO
7710 LD SP,#80
7715 JP 800T

```

```

5 *C-
10 :*****
15 :*
20 :* C O O U T *
25 :*
30 :*****
35
40 :podprogram vystupu
45 :v modu 64 znaku na radek
50 :K01-8CS-2
55 ORG #6800
60 CONOUT LD HL,(ADRCO)
65 JP (HL)
75 ADRCO DEFW COO
80 ADDCUR DEFB 20,20
85 DISP DEFS 4
90 TSCRL DEFB 1
95 TABCHR EQU #7000

```

```

100
105 :tabulka adres podprogramu,
110 :zpracovavajicich vystup
115 :znaku
120

```

```

125 TABCON
130 DEFW CONIC :00-^e
135 DEFW CONIC :01-^A
140 DEFW CONHOME :02-^B
145 DEFW CONIC :03-^C
150 DEFW CONIC :04-^D
155 DEFW CONIC :05-^E
160 DEFW CONIC :06-^F
165 DEFW COBELL :07-^G
170 DEFW COBS :08-^H
175 DEFW COTAB :09-^I
180 DEFW COLF :0A-^J
185 DEFW CONIC :0B-^K
190 DEFW COFF :0C-^L
195 DEFW COCR :0D-^M
200 DEFW CONIC :0E-^N
205 DEFW CONIC :0F-^O
210 DEFW CONIC :10-^P
215 DEFW COROLL :11-^Q
220 DEFW COPAGE :12-^R
225 DEFW CONIC :13-^S
230 DEFW CONIC :14-^T
235 DEFW CONIC :15-^U
240 DEFW COCLLI :16-^V
245 DEFW CONIC :17-^W
250 DEFW COUP :18-^X
255 DEFW CONIC1 :19-^Y
260 DEFW CORIGH :1A-^Z
265 DEFW COAT :1B-^
270 DEFW CONIC1 :1C-^I
275 DEFW CONIC1 :1D-^J
280 DEFW COBORD :1E-^
285 DEFW COINVE :1F-^

```

```

290 TABCS
295 DEFB 32 :#80
300 DEFB 144 : a
305 DEFB 32
310 DEFB 145 : c
315 DEFB 146 : d
320 DEFB 148 : e
325 DEFB 32
330 DEFB 32
335 DEFB 32
340 DEFB 149 : i
345 DEFB 156 : u
350 DEFB 32
355 DEFB 32
360 DEFB 32
365 DEFB 150 : n
370 DEFB 151 : o
375 DEFB 32
380 DEFB 32
385 DEFB 152 : r
390 DEFB 153 : s
395 DEFB 154 : t
400 DEFB 155 : u
405 DEFB 32
410 DEFB 147 : e
415 DEFB 32
420 DEFB 157 : x
425 DEFB 158 : z
430 DEFB 32
435 DEFB 159 : -
440 DEFB 32
445 DEFB 162 :
450 DEFB 32
455 DEFB

```

```

460 DEFB 128 : A
465 DEFB 32
470 DEFB 129 : C
475 DEFB 130 : D
480 DEFB 132 : E
485 DEFB 32
490 DEFB 32
495 DEFB 32
500 DEFB 133 : I
505 DEFB 140 : U
510 DEFB 32
515 DEFB 32
520 DEFB 32
525 DEFB 134 : N
530 DEFB 135 : O
535 DEFB 32
540 DEFB 32
545 DEFB 136 : R
550 DEFB 137 : S
555 DEFB 138 : T
560 DEFB 139 : U
565 DEFB 32
570 DEFB 131 : E
575 DEFB 32
580 DEFB 141 : Y
585 DEFB 142 : Z
590 DEFB 32
595 DEFB 32
600 DEFB 32
605 DEFB 32
610 DEFB 32
615
620 COO
625 LD A,C
630 CP 32
635 JR C,COCB
640 BIT 7,A
645 JR Z,COOB
650 SUB 128+64
655 RET C
660 LD H,0
665 LD L,A
670 LD DE,TABCS
675 ADD HL,DE
680 LD A,(HL)
685 COOB
690 PUSH AF
695 LD A,1
700 LD (DISP+3),A
705 POP AF
710 CALL COOPR
715 XOR A
720 LD (DISP+3),A
725 LD HL,(ADDCUR)
730 INC HL
735 LD A,64
740 CP L
745 JR NZ,COO1
750 INC H
755 LD L,0
760 COO1
765 LD A,24
770 CP H
775 CALL Z,SCROLL
780 LD (ADDCUR),HL
785 COO2
790 COOCUR
795 LD A,143
800 CALL COOPR
805 RET
810
815 :podprogramy, zpracovavajici
820 :ridici znaky
825 COCB
830 ADD A,A
835 LD L,A
840 LD H,0
845 LD DE,TABCON
850 ADD HL,DE
855 LD E,(HL)
860 INC HL
865 LD D,(HL)
870 EX DE,HL
875 JP (HL)
880 CONIC
885 LD HL,COO
890 CONIC0
895 LD (ADRCO),HL
900 RET
905 CONIC1
910 LD HL,CONIC
915 JR CONIC0
920
925 :
930 COHOME
935 CALL COOCUR
940 COHOMO
945 LD HL,0
950 LD (ADDCUR),HL
955 JR COOCUR
960 :
965 COBELL
970 LD HL,130
975 LD DE,32

```

```

980 CALL 949
985 RET
990
995 COBS
1000 CALL COOCUR
1005 LD HL,(ADDCUR)
1010 LD A,L
1015 OR H
1020 JR Z,COOCUR
1025 DEC HL
1030 LD A,FF
1035 CP L
1040 JR NZ,COO2
1045 LD L,63
1050 JR COO2
1055
1060 COTAB
1065 CALL COOCUR
1070 LD HL,(ADDCUR)
1075 LD A,L
1080 ADD A,9
1085 AND #F8
1090 LD L,A
1095 CP 64
1100 JR NZ,COO2
1105 INC H
1110 LD L,0
1115 JR COO2
1120
1125 COLF
1130 CALL COOCUR
1135 LD HL,(ADDCUR)
1140 INC H
1145 JR COO1
1150
1155 COFF
1160 LD HL,#4000
1165 DE,#4001
1170 XOR A
1175 LD (HL),A
1180 LD BC,24*32+8-1
1185 LDIR
1190 JR COHOMO
1195
1200 COCR
1205 CALL COOCUR
1210 LD HL,(ADDCUR)
1215 LD L,0
1220 JR COO2
1225
1230 COROLL
1235 LD A,1
1240 COO5
1245 LD (TSCRL),A
1250 LD RET
1255
1260 COPAGE
1265 XOR A
1270 JR COO5
1275
1280 COCLLI
1285 LD HL,(ADDCUR)
1290 LD PUSH HL
1295 COCLLO
1300 INC HL
1305 LD A,64
1310 CP L
1315 JR Z,COCLLI
1320 LD (ADDCUR),HL
1325 PUSH HL
1330 LD C,32
1335 CALL COO
1340 POP HL
1345 JR COCLLO
1350 COCLLI
1355 XOR A
1360 LD (DISP+3),A
1365 CALL COOCUR
1370 POP HL
1375 LD (ADDCUR),HL
1380 RET
1385
1390 COUP
1395 CALL COOCUR
1400 LD HL,(ADDCUR)
1405 DEC H
1410 LD A,FF
1415 CP H
1420 JR NZ,COUP0
1425 INC H
1430 COUP0
1435 JR COO2
1440
1445 CORIGH
1450 CALL COOCUR
1455 LD HL,(ADDCUR)
1460 INC HL
1465 LD A,64
1470 CP L
1475 JR NZ,CORIG0
1480 INC H
1485 LD L,0
1490 LD A,24
1495 CP H

```


| | | | | | | | | | |
|------|-----------------------------|-------------|------|--------|------------|------|-----------------------------|-------------|--|
| 1500 | JR | NZ,CORIG0 | 1740 | COOPR | | 1980 | INC | H | |
| 1505 | LD | L,63 | 1745 | PUSH | AF | 1985 | INC | DE | |
| 1510 | DEC | H | 1750 | CALL | TRANCO | 1990 | DJNZ | COOPR1 | |
| 1515 | CORIG0 | | 1755 | POP | AF | 1995 | RET | | |
| 1520 | JP | CO02 | 1760 | SUB | 32 | 2000 | | | |
| 1525 | : | | 1765 | LD | L,A | 2005 | ;podprogram prevodu adresy | | |
| 1530 | COAT | | 1770 | LD | H,0 | 2010 | ;tisknuteho znaku | | |
| 1535 | CALL | COOCUR | 1775 | ADD | HL,HL | 2015 | | | |
| 1540 | LD | HL,COAT1 | 1780 | ADD | HL,HL | 2020 | TRANCO | | |
| 1545 | COAT0 | | 1785 | ADD | HL,HL | 2025 | LD | HL,(ADDCUR) | |
| 1550 | JP | CONIC0 | 1790 | EX | DE,HL | 2030 | LD | A,H | |
| 1555 | COAT1 | | 1795 | LD | HL,TABCHR | 2035 | AND | #07 | |
| 1560 | LD | HL,(ADDCUR) | 1800 | ADD | HL,DE | 2040 | RRCA | | |
| 1565 | CP | 24 | 1805 | EX | DE,HL | 2045 | RRCA | | |
| 1570 | JR | C,COAT10 | 1810 | LD | HL,(DISP) | 2050 | RRCA | | |
| 1575 | LD | A,23 | 1815 | LD | A,(DISP+2) | 2055 | SRA | L | |
| 1580 | COAT10 | | 1820 | LD | C,A | 2060 | PUSH | AF | |
| 1585 | LD | H,A | 1825 | LD | B,8 | 2065 | ADD | A,L | |
| 1590 | LD | (ADDCUR),HL | 1830 | COOPR1 | | 2070 | LD | L,A | |
| 1595 | LD | HL,COAT2 | 1835 | LD | A,(DE) | 2075 | LD | A,H | |
| 1600 | JR | COAT0 | 1840 | PUSH | AF | 2080 | AND | #F8 | |
| 1605 | COAT2 | | 1845 | BIT | 0,C | 2085 | OR | #40 | |
| 1610 | LD | HL,(ADDCUR) | 1850 | JR | NZ,COOPR2 | 2090 | LD | H,A | |
| 1615 | CP | 64 | 1855 | LD | A,(DISP+3) | 2095 | LD | (DISP),HL | |
| 1620 | JR | C,COAT20 | 1860 | AND | A | 2100 | POP | AF | |
| 1625 | LD | A,63 | 1865 | JR | Z,COOPR3 | 2105 | LD | A,0 | |
| 1630 | COAT20 | | 1870 | LD | A,#0F | 2110 | RLA | | |
| 1635 | LD | L,A | 1875 | AND | (HL) | 2115 | LD | (DISP+2),A | |
| 1640 | LD | (ADDCUR),HL | 1880 | LD | (HL),A | 2120 | RET | | |
| 1645 | CALL | COOCUR | 1885 | COOPR3 | | 2125 | | | |
| 1650 | LD | HL,CO0 | 1890 | POP | AF | 2130 | ;podprogram scrollovani pri | | |
| 1655 | JR | COAT0 | 1895 | JR | COOPR5 | 2135 | ;reziou scroll, pri rezimu | | |
| 1660 | | | 1900 | COOPR2 | | 2140 | ;page pouzve provede presun | | |
| 1665 | COBORD | | 1905 | LD | A,(DISP+3) | 2145 | ;kurzoru, v HL vraci novou | | |
| 1670 | LD | HL,COBORD | 1910 | AND | A | 2150 | ;pozici kurzoru | | |
| 1675 | JP | CONIC0 | 1915 | JR | Z,COOPR4 | 2155 | | | |
| 1680 | COBORD | | 1920 | LD | A,#F0 | 2160 | SCROLL | | |
| 1685 | LD | C,A | 1925 | AND | (HL) | 2165 | LD | A,(TSCRL) | |
| 1690 | AND | #07 | 1930 | LD | (HL),A | 2170 | OR | A | |
| 1695 | OUT | (#FE),A | 1935 | COOPR4 | | 2175 | JR | Z,SCRPA6 | |
| 1700 | JP | CONIC | 1940 | POP | AF | 2180 | PUSH | HL | |
| 1705 | : | | 1945 | RRCA | | 2185 | CALL | #0DFE | |
| 1710 | COINVE | | 1950 | RRCA | | 2190 | POP | HL | |
| 1715 | RET | | 1955 | RRCA | | 2195 | LD | H,23 | |
| 1720 | | | 1960 | RRCA | | 2200 | RET | | |
| 1725 | ;podprogram tisku znaku z r | | | 1965 | COOPR5 | 2205 | SCRPA6 | | |
| 1730 | ;A na pozici (ADDCUR) | | | 1970 | XOR | 2210 | LD | H,0 | |
| 1735 | | | 1975 | LD | (HL),A | 2215 | RET | | |

TELEVIZNY DISPLEJ DIS-84

RNDr. Stanislav Uličiansky, Komenského 51, 040 01 Košice

Alfanumerický alebo grafický displej je jednou z najdôležitejších častí mikropočítača. V súčasnosti sú na displej osobných mikropočítačov kladené už pomerne vysoké nároky. Na ich konštrukciu sa v moderných mikropočítačoch používajú takmer výlučne špecializované VLSI kontrolery, ktoré značne zjednodušujú realizáciu týchto obvodov. V našich podmienkach nemá konštruktér mikropočítača k dispozícii v súčasnosti žiaden typ takéhoto video kontrolera, preto mu ostáva jediná možnosť — zostaviť príslušné obvody z dostupných IO nižšej integrácie, hoci na to spotrebuje takmer 10x viac obvodov.

Na stránkach AR už bolo publikovaných niekoľko zapojení TV displejov rôznej zložitosti (AND-1, AND-12, 8080 MC-DI). Preto som pôvodne nemal zámer publikovať ďalší a pomerne zložitý typ zapojenia, ktoré je na hornej hranici amatérskych možností. Opakovanie tematickej úlohy MIKROKONKURZU AR'87 na konštrukciu inteligentného TV displeja ma však podnietilo k tomu, aby som predsa len spracoval popis displeja, ktorý som postavil k svojmu amatérskemu osobnému mikropočítaču.

Základné technické údaje:

Formát zobrazenia textu: 32/64 znakov
x 15 (16) riadkov.

Formát zobrazenia semigrafických znakov:

32/64 znakov x 30(32) riadkov.

Raster znaku: 8 x 8 bodov.

Repertoár znakov: 128 štandardných znakov podľa normy ASCII + 128 znakov, ktoré si môže zvoliť užívateľ. Generátor znakov je fixný v EPROM typu 2716. Pri použití EPROM typu 2732 sa rozšíri repertoár znakov až na 512.

Módy zobrazenia znakov:

- normálny znak,
- inverzný znak,
- blikajúci znak,
- znak s dvojitou šírkou,
- zatemnený znak,
- možno nastaviť 8 úrovní jasu pre každý znak.

Pre každý znak je možné navoliť ľubovoľnú kombináciu uvedených módov zobrazenia.

Typ displeja: Displej s vlastnou obrazovou pamäťou 2x 2 kB, ktorá je súčasťou adresového priestoru mikroprocesora s prioritou prístupu procesora do tejto pamäte.

Rolovanie obrazu: Je zabezpečené obvodymi TV displeja s možnosťou jemného rolovania po jednotlivých TV linkách nahor alebo nadol.

Stredovanie obrazu: Pozíciu obrazu na obrazovke TV prijímača je možno nastavovať v horizontálnom aj vertikálnom smere.

Popis zapojenia

Pre pochopenie činnosti tohoto displeja už nevystačíme len so samotnou schémou zapojenia, ako napríklad v prípade pomerne jednoduchého displeja 8080 MC-DI [5], ale je potrebné popísať aspoň najdôležitejšie prvky a funkciu použitého riešenia. Aby tento popis nezabral príliš veľa miesta, snažil som sa ho urobiť čo najstručnejším, no pritom dostatočne presným a výstižným. Presvedčil som sa však, že urobiť takýto popis pre zapojenie pozostávajúce z 54 IO už nie je len čiste technickou záležitosťou, ale na svoje si prídá aj literárne a výtvarné umenie. Čitateľom, ktorí nie sú oboznámení s princípmi činnosti TV displejov, odporúčam sa vopred oboznámiť aspoň s článkami z predchádzajúcich ročníkov AR [1–5].

Kvôli stručnosti tu neuvádzam obvyklú blokovú schému zapojenia, ale celý popis sa bude viazať na úplnú schému zapojenia, ktorá je uvedená na obr. 1 na str. 40, 41, a na ostatné doplňujúce tabuľky a obrázky.

Popisovaný displej vytvára na obrazovke raster 512 × 256 bodov. Pretože jeden znak je zobrazovaný v rasti 8 × 8 bodov, znamená to, že môžeme zobraziť 64 znakov v 32 riadkoch. To odpovedá práve rozsahu 2048 znakov na celú obrazovku. Pretože nie sú obvodmi displeja generované trvale zatemnené riadky ako napr. v displeji AND-1, pri zápise textu musíme znaky zapisovať len do každého druhého riadku, čo nám dáva celkovo 16 zobraziteľných alfanumerických riadkov. Je to síce podstatne menej riadkov ako 24 viditeľných riadkov v AND-1, avšak vďaka väčšej medzere je text na tomto displeji oveľa prehľadnejší. Hlavným účelom tohoto riešenia je však to, aby bolo možné v semigrafickom režime pokryť znakmi celú plochu obrazovky. V alfanumerickom režime nám však toto riešenie umožňuje umiestniť do oddeľovacích prázdnych riadkov zobrazenie diakritických znamienok nad veľké aj malé písmená, vďaka čomu tu máme možnosť „naučiť“ tento displej slovenčinu alebo češtinu, čo je pri rasti znaku 8 × 8 bodov obtiažne realizovateľné.

Vyššie uvedený raster zobrazenia je určený zapojením obvodov primárnej časovej základne, ktorá pozostáva z obvodov E3, D3, D4, E4, D5, E7, E8. Prvým stupňom tejto časovej základne je oscilátor, vytvorený z invertorov E3, ktorý pracuje na frekvencii 12 MHz.

Pretože táto frekvencia je už pomerne vysoká pre bežné obvody TTL, je vhodné v tomto stupni použiť obvod typu 74S04 resp. 74LS04. Za oscilátorom nasleduje kaskáda deličov 1:16, 1:16 a 1:3, ktorými je vytvorený celkový deliaci pomer 1:768. Prvé dva stupne tejto deličky je potrebné osadiť synchronnými čítačmi typu 74193. Výsledný produkt z tejto kaskády — signál AP9 s frekvenciou 15 625 kHz — slúži ako signál, z ktorého sa odvodzuje horizontálny zatemňovací a synchronizačný impulz. Druhý stupeň primárnej časovej základne je tvorený obvody 1/2 D5, E7, E8 — vytvárajú deličku 1:320, ktorá zabezpečuje tvorbu 320 TV riadkov a signálu o frekvencii 48,8 Hz pre obvody vertikálneho zatemňovania

a synchronizácie. Z celkového rastu 768 bodov × 320 riadkov spadá do viditeľnej zóny 512 bodov × 256 riadkov. To odpovedá pomerne veľkému pokrytiu plochy obrazovky TV prijímača, vďaka čomu sa objavujú problémy s ostrosťou kresby v rohoch obrazovky a je nevyhnutné zabezpečiť možnosť presného nastavenia pozície obrazu na obrazovke. Túto úlohu plnia obvody tvorby zatemňovacích a synchronizačných impulzov.

Horizontálne zatemnenie a synchronizácia sú odvodené z výstupu AP9 prvého stupňa primárnej časovej základne (AP9 = HZATZ). Invertovaný signál HZATZ je privedený na vstup posuvného registra E9 (74164), ktorý je ovládaný hodinovým signálom AQ0 resp. AQ0. Časové posunutie, ktoré je vytvárané v tomto obvode, je potrebné na zabezpečenie správneho zobrazenia v súvislosti s použitím vyrovnávacích registrov C1, C3. Z vhodného výstupu tohoto obvodu sa odoberá signál HZATP, ktorý sa využíva na vytvorenie signálu ZAT v obvode E6/6. Je taktiež privedený na vstup ďalšieho posuvného registra D9, ktorý zaisťuje možnosť nastavenia obrazu v horizontálnom smere. Toto nastavovanie sa realizuje voľbou vhodného výstupu na D9, z ktorého sa odoberá spúšťač impulzu pre MK0 C9 (1/2 74123), ktorý generuje horizontálny synchronizačný impulz o dĺžke 4,5 až 4,9 μs. Jemné nastavenie pozície obrazu v horizontálnom smere je tiež možné urobiť voľbou signálu AP3 alebo AP3 ako hodinového signálu pre posuvný register D9. Obdobným spôsobom ako pre horizontálny rozklad sa zo signálu BP8 = VZATZ vytvára v E6/6 signál ZAT a po voliteľnom spozdení obvodom D8 sa v C9 (druhá polovica obvodu) vytvára vertikálny synchronizačný impulz o dĺžke 160 μs. Horizontálny a vertikálny synchronizačný signál sa pomocou hradla EX-OR B9/3 sčítavajú do úplnej synchronizačnej zmesi SYNC, ktorá je vyvedená na špičku č. 15 konektora K4.

Primárna časová základňa sa využíva len na vytváranie zatemňovacích a synchronizačných signálov, iba signály AP0 až AP3 sa zúčastňujú na tvorbe zobrazenia znaku ako adresa bodu znaku na TV riadku, AP0 ÷ AP2 pri zobrazení normálneho znaku a AP1 ÷ AP3 pri zobrazení znaku s dvojitou šírkou. Všetky ostatné adresovacie signály, ktoré určujú adresu TV riadku v znaku, adresu znaku v riadku a stĺpci sú generované sekundárnou časovou základňou. Prvý stupeň tejto sekundárnej časovej základne, pozostávajúci z obvodov C5, D5 1/2, F5 1/2, generuje 6bitovú adresu znaku na riadku určenú signálmi AS0 až AS5. Tento čítač je inkrementovaný signálom AQ2 = AP2 pre normálny znak a signálom o polovičnej frekvencii AQ2 = AP3 pre znak s dvojitou šírkou. Toto riešenie umožňuje dynamicky kombinovať na jednom riadku normálne a široké znaky bez nutnosti zdvojsť zápis znaku pre široký znak. Ak sú všetky znaky na riadku s dvojnásobnou šírkou, potom počas trvania aktívnej časti riadku sa stačí naadresovať len prvých 32 znakov, ostatné znaky, hoci sú zapísané v obrazovke pamäti, nebudú zobrazené. Počas zatemnenej časti riadku je inkrementácia čítača blokovaná signálom HZATZ prostredníctvom hradla E6/3. Horizontálny synchronizačný impulz vynuluje tento čítač a pre každý TV riadok sa začína počítať odznova.

Druhý stupeň sekundárnej časovej základne je tvorený obvody D6 a D7. Výstupné signály BS0 — BS2 určujú adresu TV riadku v znaku a ďalšie signály BS3 — BS7 určujú 5 bitovú adresu znakového riadku (32 možných znakových riadkov). Tento čítač je inkrementovaný horizontálnym synchronizačným impulzom HSYNC v čase, keď nie je aktívny vertikálny zatemňovací impulz (VZATZ = H) a je nulovaný vertikálnym synchronizačným impulzom.

Široká paleta zobrazovacích možností, ktoré poskytuje tento displej, je zabezpečená použitým spôsobom organizácie obrazovej pamäte. Táto pamäť pozostáva z dvoch úplne samostatných 2 kB blokov pamäte RAM, ktoré sú zostavené z 8 ks IO 2114. Tieto bloky sú z pohľadu mikroprocesora organizované ako súčasť jeho operačnej pamäte. Použité adresovacie obvody umožňujú takmer ľubovoľne rozmiestniť tieto bloky v celom priestore 64 kB. Akonáhle procesor adresuje obrazovú pamäť, pripájajú sa adresové vodiče pamäti na adresovú zbernicu procesora prostredníctvom prepínačov adresovej zbernice, zostavených z obvodov typu 74157 (IO 05, B5, B6, B7). Pripojenie pamäti na datovú zbernicu procesora sa realizuje pomocou oddeľovačov typu MH3216, pričom základnou zvláštnosťou je tu to, že pre každý pamäťový blok máme samostatný oddeľovač datovej zbernice. Vďaka tomuto riešeniu nie sú datové vodiče týchto pamäťových blokov navzájom pospájané a preto sa z hľadiska vnútorných obvodov TV displeja javí obrazová pamäť ako jediný blok 2k, ale so 16 bitovou datovou zbernicou. Táto organizácia obrazovej pamäte sa potom využíva tak, že 8 bitov reprezentuje znakový bajt a druhých 8 bitov zas príznakový bajt. Preto potom nazývame pamäťový blok realizovaný obvody B1L, B1H, B2L, B2H ako obrazovú pamäť pre príznaky a pamäťový blok s obvody B3L, B3H, B4L, B4H ako obrazovú pamäť pre znaky. Každému bajtu v pamäti znakov odpovedá príznakový bajt v pamäti príznakov s adresou vyššou o 2k v rámci adresového priestoru procesora a s rovnakou adresou v rámci obvodov displeja. Pokiaľ sa majú zobrazovať znaky v rovnakom móde zobrazenia, potom na zápis znaku stačí jeden bajt, pričom predpokladáme, že pamäť príznakov je naplnená požadovaným obsahom. Pokiaľ je potrebné zapisovať znaky s rôznym módom zobrazenia, potom sa musia do obrazovej pamäti zapisovať 2 bajty — znakový aj príznakový.

Datová zbernica pamäti znakov DZX sa obvyklým spôsobom pripája na adresové vodiče A3 až A10 generátora znakov, ktorý je v pamäti EPROM typu 2716. Najnižšie tri adresové vodiče EPROM sú napojené na signály BQ0, BQ1, BQ2, ktoré reprezentujú adresu TV riadku v znaku. Na výstupoch obvodu C4 máme v paralelnej podobe prítomnú osmicu bitov, ktoré odpovedajú požadovanému zobrazeniu svetlých a tmavých bodov v znaku. Táto paralelná informácia sa neprevádza okamžite na sériový tvar, ale prechodne sa uchováva vo vyrovnávacom registri C3 (MH3212). Výstupy z obrazovej pamäte pre príznaky DPX sa pripájajú na druhý vyrovnávací register C1, opäť s IO MH3212. Tieto dva vyrovnávacie registre C1, C3 sú riadené signálom STBV tak, aby sme na ich výstupoch mali

vždy platnú informáciu pre zobrazovaný znak — vlastnú znakovú časť aj príznakovú časť. Pomocou týchto vyrovnávacích registrov sa odstraňuje spozdenie, ktoré by inak bolo spôsobené prechodom informácie o zobrazovanom znaku cez generátor znakov v pamäti EPROM. Pretože pre normálny znak a znak s dvojnásobnou šírkou je potrebné meniť obsah vyrovnávacích registrov v odlišných okamžikoch, signál STVB je vytváraný prepínačom D2 (74157), pričom pre normálny znak je STVB = AP0. AP1. AP2 (výstup F3/6) a pre široký znak je STBV = CAD3 (výstup D3/12).

Výstupy z vyrovnávacieho registra C3 sú pripojené na vstupy multiplexera C2 (74151), na ktorého adresové vstupy A, B, C sú pripojené signály AQ0 až AQ2. Tieto signály sú, ako už bolo spomínané, vytvárané zo signálov AS0 až AS3 pomocou prepínača D2 (74157). Tým je zabezpečené správne vytváranie zobrazenia bodu v znaku v závislosti na zvolenom type znaku (normálny resp. široký). Multiplexer C2 je navyše prostredníctvom vstupu S hradľovaný signálom z F3/8, čo je signál s periódou okolo 1 s vytváraný multivibrátorom E2. Samotný signál z multivibrátora je však ešte hradľovaný signálom z výstupu C1/04 prostredníctvom hradla F3/12. Tento obvod zabezpečuje funkciu blikania obrazu, ktoré je povoľované nastavením bitu D0 = H = log 1 v príznakovom bajte. Výstupné signály VIDEO a VIDEO z multiplexera C2 sú cez sústavu z hradliel D1 privedené na vstup E3 dekodéra E1, realizovaného pomocou obvodu typu MH 3205. V závislosti na stave výstupu C1/06 sa preto potom realizuje normálne alebo inverzné zobrazenie znaku. Výstupný signál C1/08, odpovedajúci bitu D2 v príznakovom bajte, je označený ako DBL a zabezpečuje prepínanie obvodu D2, teda obvodu prepínania zobrazenia normálneho znaku a širokého znaku.

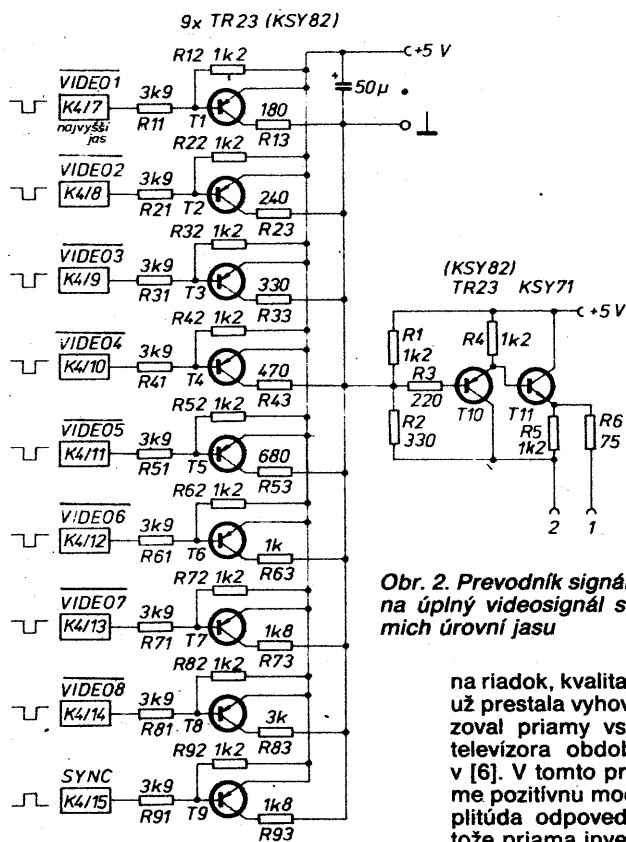
Ďalší signál C1/10 = D3 v stave D3 = H = log 1 spôsobuje, že sa prostredníctvom riadiaceho vstupu E1 zablokuje dekodér E1, vďaka čomu sa zatemňuje zobrazenie rovnako ako sa zatemňuje počas signálu ZAT = H = log 0, ktorý je privedený na povoľovací vstup E2 obvodu E1. K zatemneniu zobrazenia dochádza však taktiež počas doby, kedy je obrazová pamäť pripojená na zbernicu procesora a signál SEL (pripojený na vstup hradla A5/08) je v aktívnom stave L. Vďaka tomu sa rušenie zobrazenia na obrazovke počas zápisu dát do obrazovej pamäte realizuje ako čierne čiarky vytrhávajúce z bieleho obrazu. Ako je známe, toto rušenie sa nedá jednoduchým spôsobom odstrániť. V tejto podobe však pri bielom písme na čiernom pozadí bude minimalizované.

Adresové vstupy dekodéra E1 sú pripojené na ďalšie 3 bity (D4, D5, D6) vyrovnávacieho registra pre príznaky (obvod C1). V závislosti na stave týchto bitov sa potom pri aktivácii dekodéra prostredníctvom povoľovacích vstupov E1, E2, E3 aktivuje do stavu L jeden z výstupov 0 až 7. Na tieto výstupy je potom pripojený obvod prevodníka TV úrovni, ktorý je realizovaný tak, že každému z výstupov 0 až 7 dekodéra E1 priradí inú úroveň jasu. Schéma tohoto prevodníka je uvedená na obr. 2. Realizovaný je na malej samostatnej doske plošného spoja, ktorá je spojená s doskou displeja káblom prostredníctvom konektora K4.

Je to vlastne veľmi jednoduchý prevodník D/A rozlišujúci 8 analogových úrovní. Prvá verzia tohoto prevodníka bola spojená s vľ TV modulátorom. V tomto prípade sa vyžadovala negatívna modulácia a preto bolo vhodné, že dekodér E1 má výstupy aktívne v stave L. Celé zapojenie preto obsahovalo len 3 tranzistory, niekoľko diód a pasívne prvky. Keď som však realizoval verziu displeja so 64 znakmi

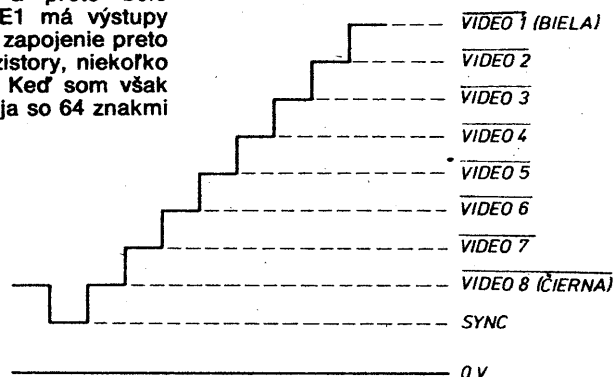
nou moduláciou v tranzistorovom stupni nedávala rozumnú možnosť nastavovania jednotlivých úrovní jasu, ukázalo sa ako jediné vhodné, aj keď trochu nákladné, riešenie s tranzistori p-n-p spínajúcimi v aktívnom stave rezistory R_{X3} ($X=1$ až 9) na napätie +5 V. V súčtovom bode x potom získame potrebné napätie s odstupňovanými amplitúdami, ako je to schématicky znázornené na obr. 3. Odstupňovanie amplitúd sa dá nastavovať zmenou rezistorov R_{X3} .

Na záver popisu zapojenia uvádzam ešte stručné údaje o ostatných obvodoch displeja. Obvody 05, B5, B6, B7 (4×74157) tvoria prepínač adresovej zbernice pre pamäť. Tento prepínač pripája pamäte na adresovú zbernicu procesora keď je signál SEL = H, tj. keď chce procesor čítať alebo zapisovať dáta do obrazovej pamäte. Súčasne tento signál prostredníctvom invertora A6/12 a hradla A5/8 zatemňuje obrazovku počas prístupu procesora do pamäte. Na vstupy X1 týchto prepínačov sú pripojené adresové vodiče BA0 až BA9 adresovej zbernice procesora. Na druhú sadu vstupov X0 sú pripojené priamo vodiče ASX prvého stupňa sekundárnej časovej základne, zatiaľ čo výstupy BSX druhého stupňa časovej základne idú najprv do sumátora, realizovaného obvodmi C6, C7, a až odtiaľ ako signály BQX idú na vstupy prepínača adres. V sumátore sa realizuje súčet dvoch 8 bitových hodnôt MODULO 256, kde druhou vstupnou veličinou je obsah registra C8. Pokiaľ je v registri C8 zapísaná hodnota 00H, potom sa nič nedeje. Keď však do neho zapíšeme hodnotu napríklad 0FH, potom v okamžiku, keď sa na obrazovke



Obr. 2. Prevodník signálov video a sync na úplný videosignál s rozlíšením osmich úrovní jasu

na riadok, kvalita obrazu cez modulátor už prestala vyhovovať. Preto som realizoval priamy vstup video signálu do televízora obdobným spôsobom ako v [6]. V tomto prípade však potrebujeme pozitívnu moduláciu (najväčšia amplitúda odpovedá úrovni bielej). Pretože priama inverzia signálu s negatív-



Obr. 3. Schématické znázornenie vzťahov medzi amplitúdami výstupného signálu z prevodníka TV úrovne pre jednotlivé signály VIDEO X a SYNC

začína písať prvý aktívny televízny riadok, obrazová pamäť a generátor znakov je adresovaný tak, že sa z nich vyberajú údaje, ktoré odpovedajú šiestému TV riadku. Inkrementáciou registra C8 o +1 sa posúva obraz o 1 TV riadok nahor, dekrementáciou sa posúva o 1 TV riadok nadol. Cyklickým opakováním tejto inkrementácie alebo dekrementácie sa realizuje takmer spojité posúvanie obrazu smerom nahor alebo nadol, ktoré je veľmi prljemné na oči a znižuje zrakovú únavu pri dlhšetrávajúcej práci s počítačom. Obvyklé rolovanie obrazu o jeden znakový riadok pri príchode znakov CR LF sa potom realizuje ako cyklus 16 inkrementácií v smyčke s časovým spozdením, ktorého zmenou sa nastavuje požadovaná rýchlosť rolo-

vania. Pracujem s profesionálnym počítačom s displejom typu VT 100, ktorý má realizované toto spojité rolovanie a preto viem oceniť jeho výhody.

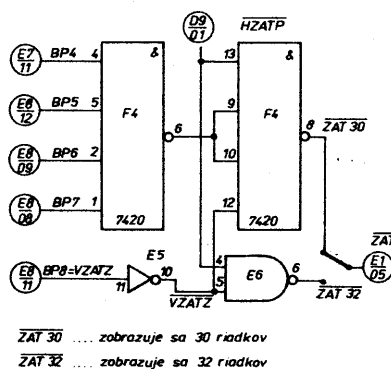
Výstupný signál BQ7 zo sumátora nie je využívaný ako adresový vodič, ale zabezpečuje prepínanie horného a dolného bloku obrazovej pamäte znakov a príznakov. K tomuto prepínaní musí dochádzať v okamžiku, keď sa ukončí kreslenie 127 riadku a má sa začať kresliť 128 TV riadok. K zmene hodnoty BS7 = BQ7 z L na H dochádza pri nábežnej hrane signálu HSYNC, ktorý inkrementuje čítače D6, D7. Aby nedochádzalo ku kolízii medzi dolnou a hornou časťou obrazovej pamäte pri prepínaní, je potrebné zaistiť, aby na určitú dobu a vo vhodnom okamžiku boli obidve pamäte odpojené. Dosahuje sa to tým, že signály BQ7 a BQ7 sú počas doby horizontálneho synchronizačného impulzu (4,5 μ s) zahradlované v obvodoch F7/6, F7/11 signálom HSYNC.

Zapojenie obvodov dekodéra adres je v podstate štandardné. Je zabezpečené úplné dekodovanie adres, pričom je možné zvoliť uloženie bloku 4 kB obrazovej pamäte ľubovoľne v rámci každého z ôsmich blokov 8 kB. Jedinou zvláštnosťou dekodéra adres je to, že vďaka použitiu samostatných oddeľovačov adresovej zbernice pre pamäť znakov a pamäť príznakov bolo nevyhnutné zdvojiť obvody generácie signálov CS a DCE pre oddeľovače s IO typu 3216. V zapojení, ktoré je uvedené na obr. 1, je obrazová pamäť pre znaky uložená na adresách E000H až E7FFH a pamäť pre príznaky na adresách E800H až EFFFH. Nakoľko je použité rozdelenie obrazu na riadky a stĺpce v mocninách čísla 2 (64 \times 32), je zostavenie tabuľky adres pre jednotlivé pozície na obrazovke jednoduché a nie je potrebné ho zvlášť uvádzať.

Pri zobrazovaní 256 TV riadkov z celkového počtu 320 je pokrytá takmer celá plocha obrazovky, pričom v rohoch obrazovky sa stávajú písmená vďaka astigmatizmu nečitateľné. Tento problém je zvlášť citelný, keď chceme použiť zobrazenie so 64 znakmi na riadok. Preto som realizoval doplnujúci obvod, ktorý predlžuje vertikálny zatemňovací impulz tak, že sú zatemnené tiež posledné dva (30. a 31.) riadky. Po odpovedajúcej zmene polohy vertikálneho synchronizačného impulzu — vystredení obrazu — je už čitateľnosť lepšia. Zapojenie tohoto obvodu je uvedené na obr. 4. Pokiaľ nepoužívame TV prijímač aj na bežné účely, je možné tento problém vyriešiť jednoducho zmenšením vertikálneho rozmeru obrazu prestavením príslušného ovládacieho prvku na TV prijímači. Vytvorením zatemnenej časti aktívnej zóny obrazu však je možné docieľať niektoré zvláštne efekty. V tejto zatemnenej zóne sa môže realizovať nulovanie riadku alebo zápis nového obsahu riadku, vďaka čomu sa horizontálna zmena obsahu riadku stane neviditeľná a pri spojitom rolovaní textu sa na dolnej hrane obrazu bude vynárať už hotový zapísaný riadok. Taktiež je možné realizovať rolovanie obrazu o jeden riadok (jeden dvojriadok) späť s minimálnou programovou podporou.

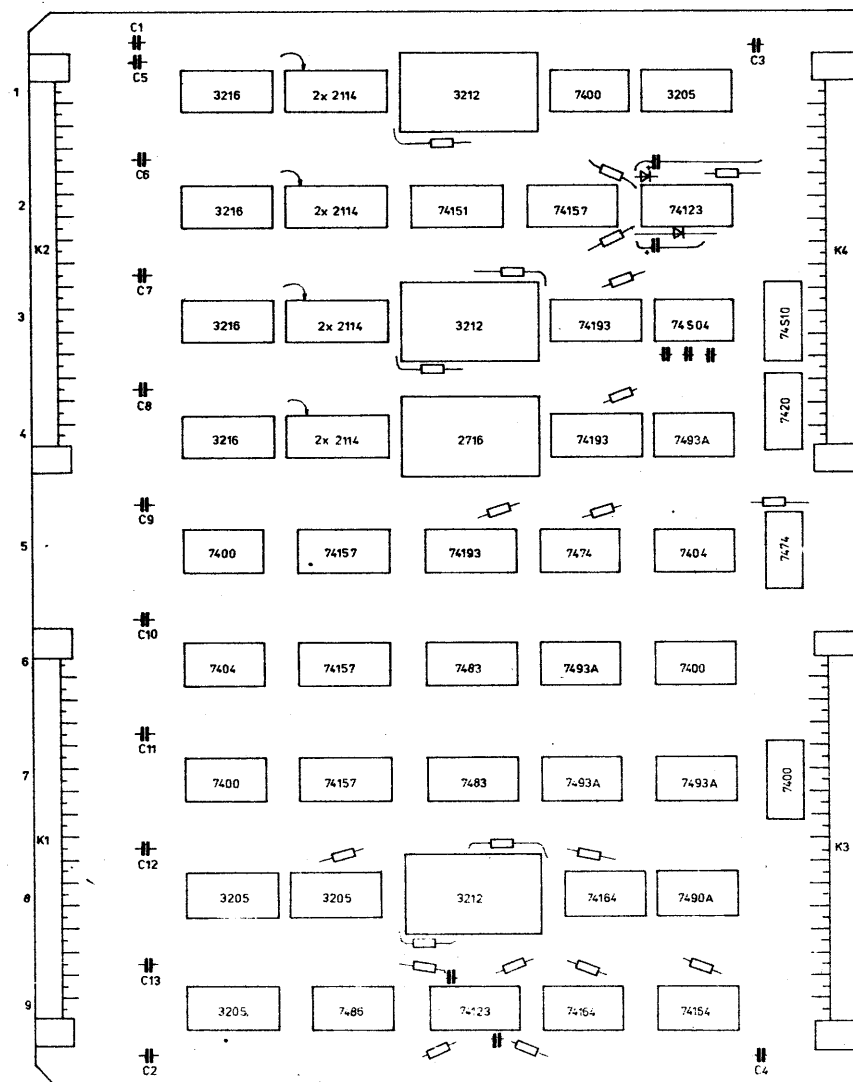
Konstrukcia displeja a prevodníka TV úrovni

Displej bol realizovaný ako súčasť amatérskeho počítača TOM 84 na univerzálnej doske plošného spoja veľkého európskeho formátu o rozmeroch 235 \times 180 mm. Spojový obrazec tejto dosky bol uverejnený v ročenke AR'87 — MIKROELEKTRONIKA (typ V 215 ME). Osadenie tejto dosky integrovanými obvodmi a ďalšími súčiast-



Obr. 4. Doplnkový obvod, ktorý zabezpečuje predĺženie zatemnenia tak, že je zatemnený aj 30. a 31. riadok

kami je uvedené na obr. 5. Sú použité konektory typu 6AF89654 (31 kolíkové, pozlátené), pričom konektory K1 a K2 sú pre systémovú zbernicu mikropočítača, na K4 sú vyvedené výstupy VIDEO a SYNC a K3 je zatiaľ nevyužitý. Na tento konektor je možné vyviesť výstupy z generátora znakov (obvod C4) a potom nahradiť pamäť EPROM pripojením na simulátor EPROM, čo dovoľí dynamicky meniť súbor zobrazovaných znakov pomocou programu. Pôvodne sa na doske rátalo s pamäťou 2 \times 1k, čo vystačí na formát 32 \times 32 znakov. Keď som realizoval úpravu na formát 64 \times 32 znakov, nezostalo mi už iné riešenie než umiestniť ďalšie pamäte na „poschodie“ opatrným spájaním dvojíc obvodov s medzerou asi 2 mm medzi ich púzdrami, aby sa aspoň trochu zabezpečilo ich chladenie. Iba signály CS nie je možné privádzať ku dvom púzdram paralelne, ale je potrebné na hornom z ich opatrne odhnuť nožičku a CS pripojiť drôtkom. Takto vytvorená dvojica je zasunutá v päťici, vyrobenej z dostupných 16 vývodových päťíc rozrezaním a zlepením potrebného počtu kontaktov. Toto netradičné riešenie som si mohol dovoliť kvôli tomu, že v počítači je umiestnený malý ventilátor, ktorý sa stará aspoň čiastočne o chladenie systému. Displej má v tejto podobe už „odjazdených“ niekoľko desiatok hodín a zatiaľ sa



Obr. 5. Rozloženie súčiastok displeja DIS-84 na univerzálnej doske s plošnými spoji V215ME (medzi konektormi K1 a K2 chýba obvod 74157)

neobjavili žiadne problémy s pamätami RAM. Trošku problémov robí päťica pre pamäť EPROM, kde sa občas objavujú bodové poruchy zobrazovania znakov (nadbytočné biele bodky v znaku), ktoré sú spôsobené nedokonalým kontaktom pamäte EPROM v päťici a zmiznú po ich prečistení.

Obvody 05, F3, F4, F5, F7 sú umiestnené mimo hlavné spojové pole. Potrebne plošky pre umiestnenie týchto obvodov sa získali úpravou pomocných plošiek medzi konektormi (pre 05 a F5), prípadne plošiek, na ktoré sa pripájajú samotné konektory (pre ostatné doplnené obvody).

Obvod prevodníka signálov VIDEO a SYNC na úplný TV signál je realizovaný na malej dosičke jednostranného plošného spoja. Presný obrazec spojovej dosky neuvádzam, nakoľko by bolo vhodné urobiť jeho úpravu nahradením pevných rezistorov R_{23} trimrami, čo umožní presnejšie nastavovať jasové úrovne pre jednotlivé stupne jasu. Keďže sa jedná o pomerne jednoduché zapojenie, takýto návrh zvládne každý.

Napriek tomu, že sa jedná o pomerne zložité zapojenie, pri jeho oživovaní vystačíme s logickou sondou a osciloskopom, pomocou ktorého nastavíme oscilátor na požadovaných 12 MHz a postupne kontrolujeme jednotlivé priebehy na výstupoch primárnej a sekundárnej časovej základne. Rovnako skontrolujeme a nastavíme požadovanú dĺžku pre impulzy HSYNC a VSYNC. Pokiaľ nemáme generátor znakov v pamäti EPROM, možno funkciu displeja skúšať tak, že jednotlivé výstupy v päťici pre EPROM pripojíme postupne na zem krátkym drôtkom, pričom sa nám majú vytvárať na obrazovke čierne-biele zvislé pásiky usporiadané podľa toho, ktoré výstupy máme uzemnené. Pamäte RAM osadzujeme až po dokonalej preskúšaní funkcie dekodéra adres, oddeľovača datovej zbernice a prepínača adresovej zbernice. Pri oživovaní má displej voči iným obvodom obrovskú výhodu v tom, že sa nám činnosť displeja veľmi názorne prejavuje na obrazovke, čo veľmi pomáha pri hľadaní prípadných závad. Pri precíznej práci pri zapájaní dosky môže táto pracovať takmer na prvé zapnutie.

Programová obsluha displeja DIS 84

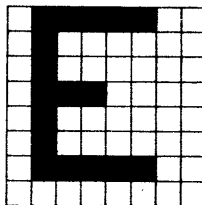
Napriek tomu, že tento displej je zložitejší než napr. AND-1, jeho programovanie sa nekomplikuje, naopak v niektorých ohľadoch sa stáva jednoduchším.

Prvou programátorskou úlohou pri oživovaní tohoto displeja je naprogramovanie generátora znakov, ktorý je v pamäti EPROM typu 2716. Táto pamäť nám pri rastrovi 8×8 bodov na jeden znak dovoľuje vytvoriť až 256 znakov. Pritom je využitá do posledného bitu. Prvých 128 znakov s D7 = 1 odpovedá súboru ASCII znakov. Namiesto riadiacich nezobraziteľných kódov možno zadefinovať rôzne semi-grafické symboly.

Spôsob naprogramovania znaku je zrejmy z príkladu uvedeného na obr. 6 pre znak E. V displeji 8080 MC-DI bol použitý rovnaký spôsob kódovania znakov v generátore znakov s EPROM typu 2708 až na inú polohu znaku 5×7 v rastrovi 8×8 . Je preto možné použiť tabuľku znakov vytvorenú pre tento displej aj pre displej DIS-84.

ADRESA

228H
229H
22AH
22BH
22CH
22DH
22EH
22FH



KÓDY

7CH
40H
40H
70H
40H
40H
7CH
00H

Obr. 6. Príklad kódovania znaku v generátore znakov

Druhú sadu znakov s D7 = 1 si môže navrhnuť každý sám podľa svojich požiadaviek. Možno sem umiestniť napríklad celú grécku abecedu a rôzne symboly používané vo vedeckotechnických textoch, ale tiež rôzne grafické obrazce, ktoré sa dajú využívať na vytváranie obrazov a na hry. Kto má tlačiareň, ten sa bude snažiť prispôbiť tieto znaky tomu, čo vie jeho tlačiareň.

Pre displej DIS-84 však môžeme vytvoriť tiež generátor znakov s 512 symbolmi pri použití pamäte typu 2372, pričom adresový vodič A11 tejto pamäte je napojený na bit D7 príznakového bajtu (viď obr. 1.). To nám umožňuje programovým nastavením tohoto bitu ovládať voľbu dvoch súborov po 256 znakov. V súbore riadiacich kódov ASCII je dvojica SO, SI, ktorú možno použiť na takéto prepínanie.

Z vyššie uvedeného vyplýva, že celý znakový bajt sa využíva na voľbu znaku. Mód zobrazovania znaku je ovládaný príznakovým bajtom. V tabuľke č. 1 je uvedené priradenie módov zobrazovania jednotlivým bitom príznakového bajtu, v tabuľke č. 2 sú zas uvedené kódy pre všetky možné kombinácie módov zobrazovania.

V obvyklom alfanumerickom režime sa pri nulovaní displeja musí zabezpečiť zapísanie celej obrazovej pamäte príznakom zvoleným kódom módu zobrazovania, napríklad 00H pre 64 znakov na riadok v normálnom zobrazení, 04H pre 32 znakov na riadok v normálnom zobrazení. Realizácia kurzora je veľmi jednoduchá a spočíva v zápise kódu pre inverzný a blikajúci znak do príslušného miesta pamäte príznakov.

Programová obsluha rolovania obrazu je zrejmá z toho, čo už bolo uvedené v popise zapojenia.

Pre zabezpečenie činnosti displeja v bežnom alfanumerickom režime už mám vypracovaný program, ktorý vznikol úpravou programového modulu OUTDIS napísaného pre displej AND-1. Nebol však zatiaľ preskúšaný v rámci celého monitora mikropočítača TOM-84, na ktorom ešte stále pracujem, preto jeho výpis zatiaľ nepripájam. Semigrafické možnosti displeja sú také bohaté, že začína byť problémom ich všetky plne využívať. Nakoľko však je výber znakov a módov zobrazovania realizovaný nastavením jednotlivých bitov v znakovom a príznakovom bajte, je ich ovládanie pomerne jednoduché a dá sa realizovať ako nastavovanie a nulovanie programových masiek pre obidva bajty.

Úplne na záver uvádzam ešte jeden námet, určený pre majiteľov farebných televizorov. Bity D4 — D6 príznakového bajtu sa využívajú na ovládanie jasu zobrazovania znaku. Je však možné zmeniť zapojenie tak, že sa využijú na ovládanie farby. Potrebnu úpravu zapojenia neuvádzam, pretože som sa touto otázkou zatiaľ nezaoberal.

Tabuľka č. 1. Význam bitov príznakového bajtu

D0 — Blikanie
D1 — Inverzia
D2 — Dvojitá šírka znaku
D3 — Blokovanie zobrazenia (programové zatemnenie)
D4 — Jas A (Farba R)
D5 — Jas B (Farba G)
D6 — Jas C (Farba B)
D7 — Prepínanie súboru znakov v EPROM 2732

Tabuľka č. 2. Kódy pre programovanie módov činnosti displeja

X0H 64 znakov na riadok, normálne zobrazenie
X1H 64 znakov na riadok, blikajúci znak
X2H 64 znakov na riadok, inverzné zobrazenie (biele pozadie)
X3H 64 znakov na riadok, inverzný a blikajúci znak
X4H 32 znakov na riadok, normálne zobrazenie
X5H 32 znakov na riadok, blikajúci znak
X6H 32 znakov na riadok, inverzné zobrazenie
X7H 32 znakov na riadok, inverzný a blikajúci znak
X8H až XFH — zatemnenie zobrazenia

0XH maximálny jas znaku

7XH minimálny jas znaku

8XH až FXH — navolená druhá sada 256 znakov v EPROM 2732

Výsledný kód príznakového bajtu sa získava lineárnou kombináciou uvedených možností (X je ľubovoľná hodnota 0 až F).

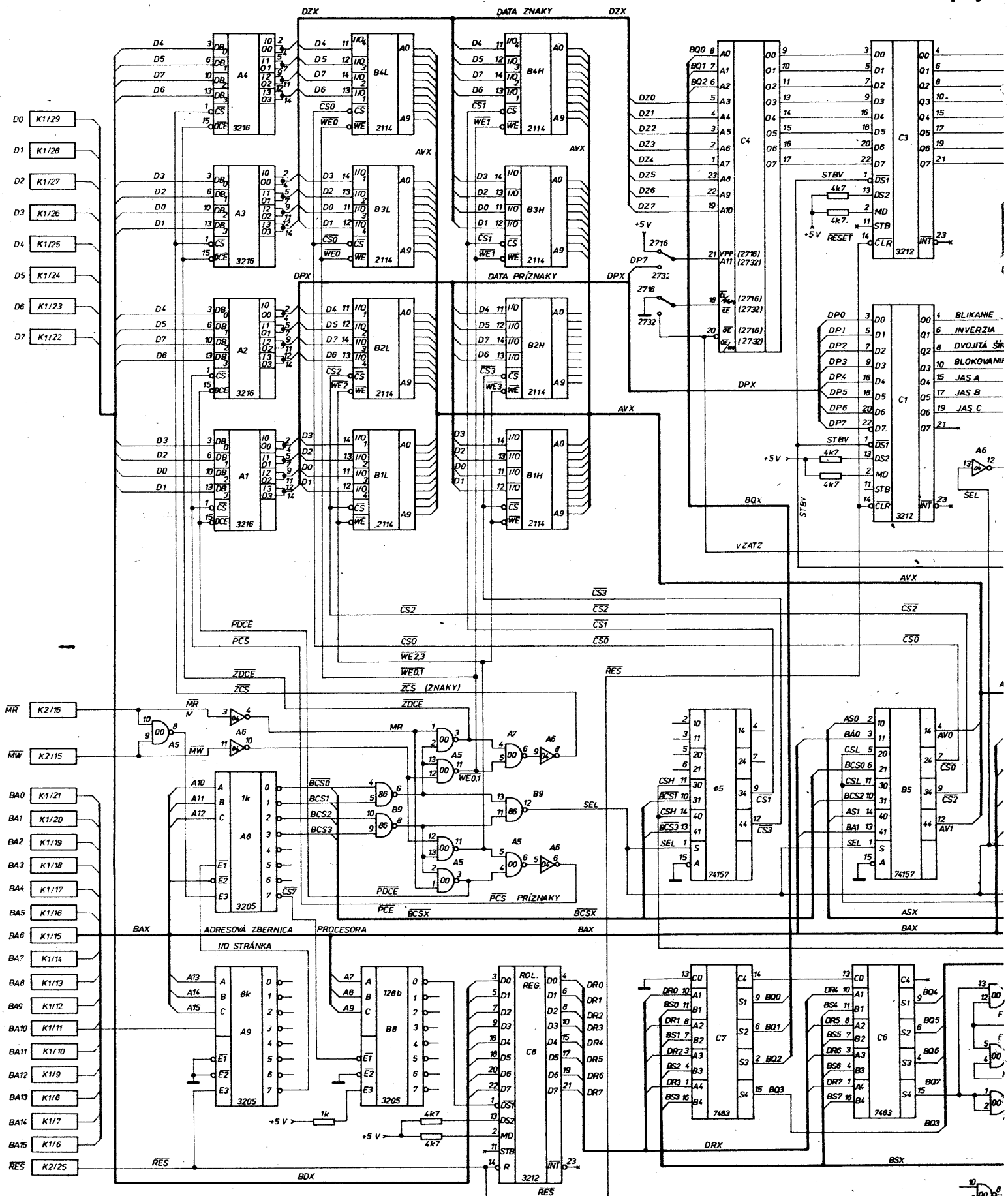
Tabuľka č. 3. Zapojenie konektorov K1, K2, K4

| Kontakt | K1 | K2 | K4 |
|---------|-------|-----------|---------|
| 1 | zem | zem | zem |
| 2 | +5 V | +5 V | +5 V |
| 3 | INT 3 | zem | — |
| 4 | INT 2 | -5 V | — |
| 5 | INT 1 | zem | — |
| 6 | A15 | +12 V | — |
| 7 | A14 | zem | VIDEO 1 |
| 8 | A13 | +15 V | VIDEO 2 |
| 9 | A12 | -15 V | VIDEO 3 |
| 10 | A11 | PIP | VIDEO 4 |
| 11 | A10 | CS0 (KBD) | VIDEO 5 |
| 12 | A9 | CS1 (KBD) | VIDEO 6 |
| 13 | A8 | IOW | VIDEO 7 |
| 14 | A7 | IOR | VIDEO 8 |
| 15 | A6 | MW | SYNC |
| 16 | A5 | MR | — |
| 17 | A4 | INTA | — |
| 18 | A3 | M1 | — |
| 19 | A2 | AEN | — |
| 20 | A1 | HLDA | — |
| 21 | A0 | DEN | — |
| 22 | D7 | STSTB | — |
| 23 | D6 | HOLD | — |
| 24 | D5 | INTE | — |
| 25 | D4 | RES | — |
| 26 | D3 | OSC | — |
| 27 | D2 | RTL | — |
| 28 | D1 | RDY | — |
| 29 | D0 | φ2 TTL | — |
| 30 | +5 V | +5 V | +5 V |
| 31 | zem | zem | zem |

Literatura:

- [1] Peterka, L.: Televizní displej I. AR/A 3/1984, str. 97—99.
- [2] Peterka, L.: Televizní displej II. AR/A 4/1984, str. 137—139.
- [3] Smutný, E.: Alfanumerický displej AND-1. AR/B 2/1983, str. 51—60.
- [4] Smutný, E.: Displej AND-1Z. AR/B 6/1985, str. 221—228.
- [5] Sojka, E.: Televizní displej 8080 MC-DI. AR/A 5/1987, str. 177—181.
- [6] Tůma, M.: TV monitor k ZX Spectru. AR/A 8/1987, str. 304

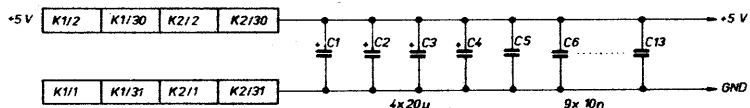
Obr. 1. Schéma
neho displeja I

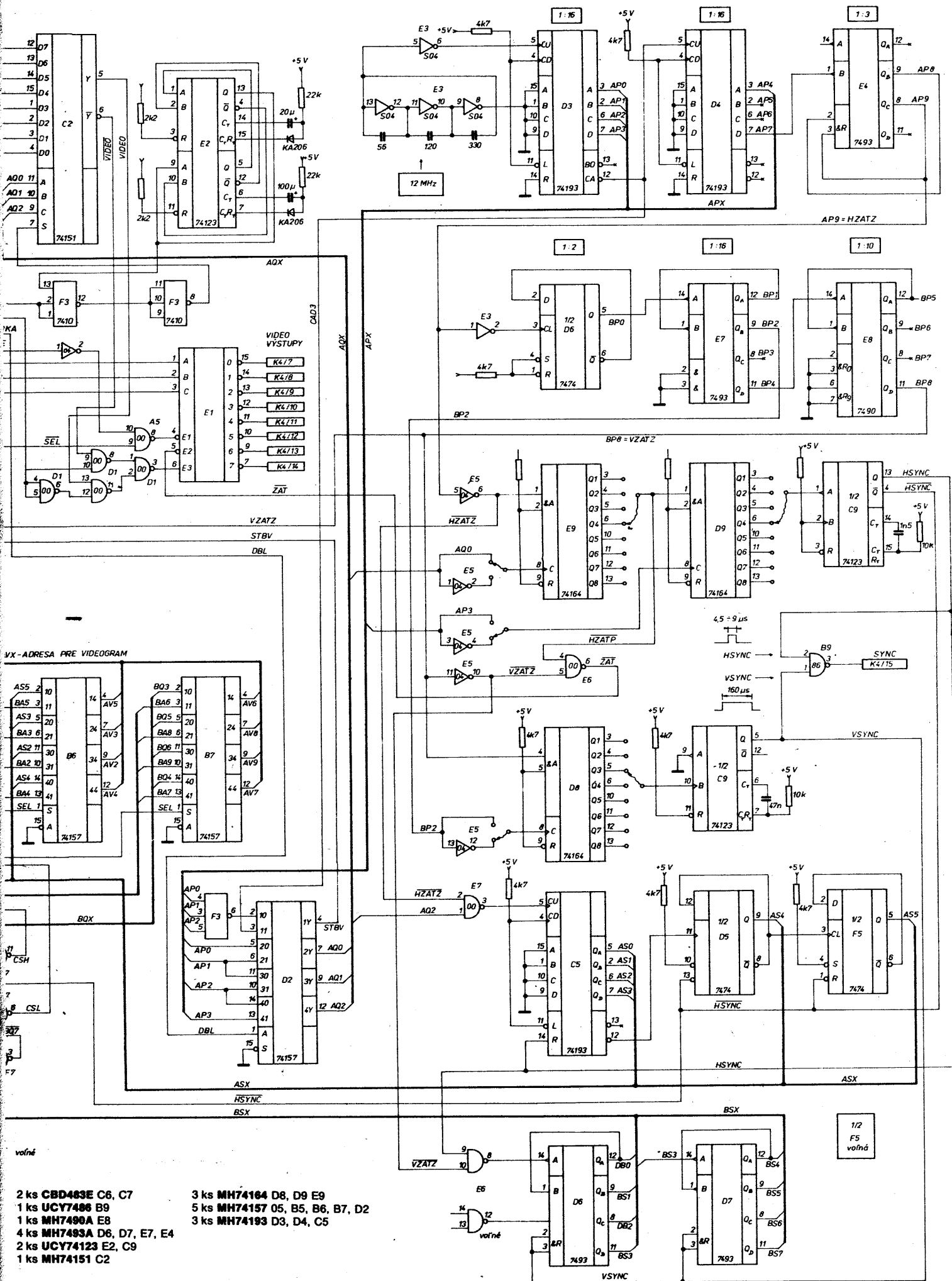


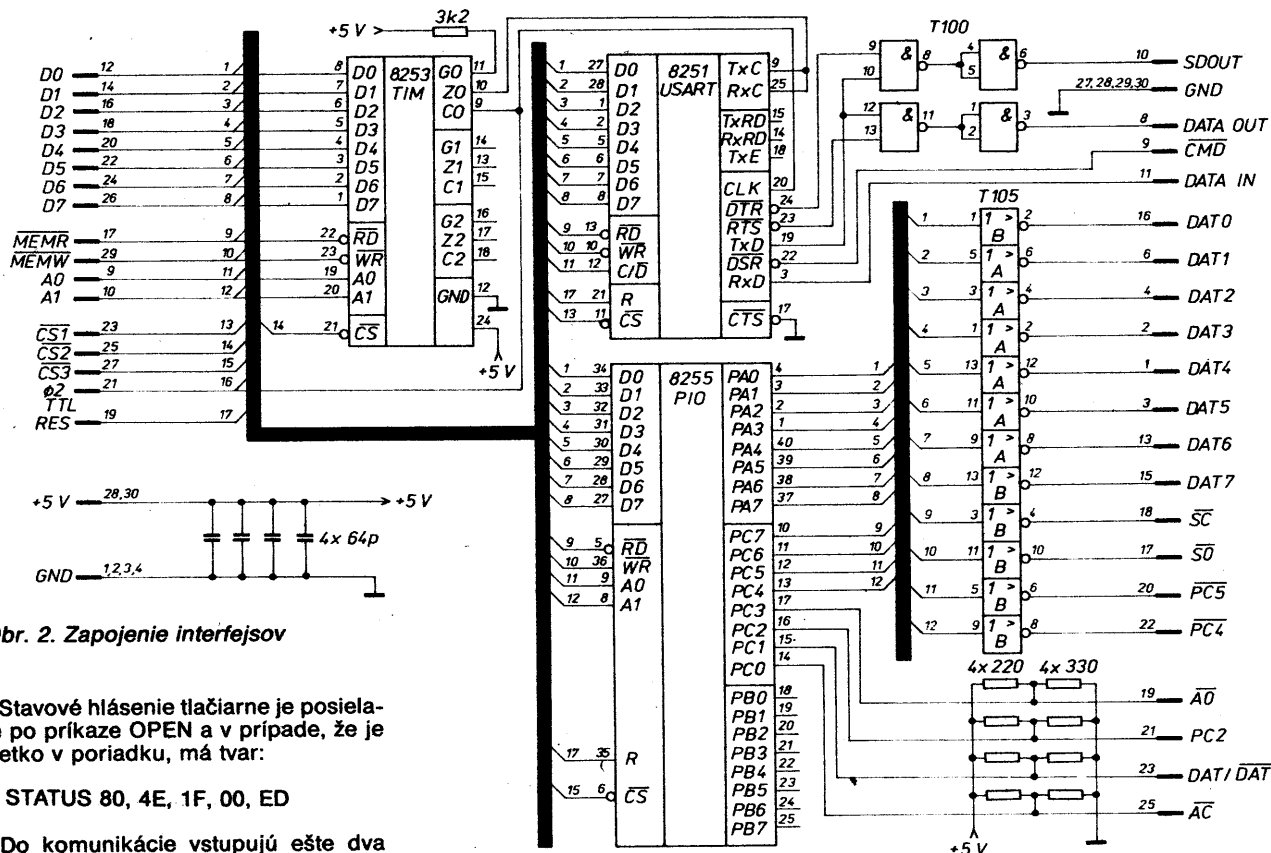
Zoznam použitých súčiastok

8 ks MHB2114 B1L, B1H, B2L, B2H, B3L, B3H,
B4L, B4H
1 ks 2716 (2732) C4
4 ks MH3205 A8, A9, B8, E1
3 ks MH3212 C1, C3, C8
4 ks MH3216 A1, A2, A3, A4

5 ks MH7400 A5, A7, D1, E6, F7
2 ks MH7404 A6, E5
1 ks MH7404 E3
1 ks MH7410 F3
1 ks MH7420 F4
2 ks MH7474 D5, F5







Obr. 2. Zapojenie interfejsov

Stavové hlásenie tlačiarne je posiela-
né po príkaze OPEN a v prípade, že je
všetko v poriadku, má tvar:

c) STATUS 80, 4E, 1F, 00, ED

Do komunikácie vstupujú ešte dva
špeciálne znaky, ktoré som nazval:

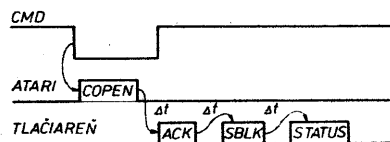
- d) ACK 41 — tlačiareň potvrdí príjem
- e) SBLK 43 — po tomto znaku tlačiareň
vyšle svoje stavové hlásenie.

Datový blok

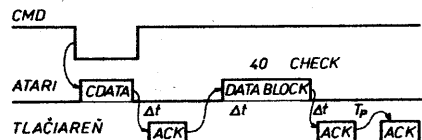
Datový blok je štandardne dlhý 40
znakov a je doplnený 41. znakom, čo je
opäť kontrolná suma. ATARI nevyšle
nikdy datový blok inej dĺžky aj keby mal
tento obsahovať čo len jeden platný
znak. Dĺžka datového bloku je určená
parametrom v operačnom systéme.

V datovom bloku má výnimočné
postavenie znak 9B, ktorý je možné
chápať ako znak ukončujúci dáta
— EOT resp. EOF — nakoľko sa ním
doplní datový blok do definovanej dĺžky
(40) pri príkaze CLOSE. Príkaz CLOSE
spôsobí výstup bežného datového blo-
ku, ktorého obsahom je zbytok vyro-
vnávajúcej pamäte doplnený resp.
ukončený znakom 9B. Riadiaci signál
CMD je aktívny pri vysiadaní riadiaceho
príkazového bloku.

Grafické znázornenie komuni- kačného protokolu



Obr. 3a. Interpretácia príkazu Open n,
code, aux, "P"



Obr. 3b. Vytlačenie naplnenej vyro-
vnávacej pamäte resp. interpretácia
CLOSE

Definovať presné časové limity,
znázornené na obrázku ako Δt , sa mi
nepodarilo, ale tento čas je zhora
obmedzený tak, aby programové vyba-
venie partnerského mikropočítača stih-
lo bez problémov vyhodnotiť prijatú
informáciu a vyslať odpoveď. Rádove
sú to ms a operačný systém ATARI pri
prekročení niektorého z týchto časov
hlási chybový oznam.

Výnimočne dlhý je čas T_p , ktorý by
bolo možné uviesť v sekundách. Tento
čas je využitý na vytlačenie prijatého
datového bloku a vyhovuje aj najpo-
malším tlačiarňam bez vyrovnávacích
pamätí.

Pri pohľade na takto definovaný
interfejs tento budí dojem jednoducho-
sti a účelnosti. Je na škodu veci, že ho
firma ATARI nepublikuje. Pri analýze
jednotlivých znakov v riadiacich štruk-
túrach je možné zistiť ich význam.
Z hľadiska emulácie tlačiarne, ktorá
nepozná iný stav ako READY
a v každom inom prípade narušuje
protokol, je význam týchto znakov
nepodstatný až na jeden, a to je adresa.
Z porovnania riadiacich štruktúr
a z pokusných zapojení som určil, že je
vhodné reagovať na prvý znak ako na
adresu, čo je v prípade tlačiarne 40H.

Úplný obraz emulovaného ATARI
interfejsu poskytuje výpis zdrojového
textu ovládacieho programu zapísan-
ého v assembleri 8080.

POPIS TECHNICKÉHO VYBAVENIA ADAPTÉRA ATARI INTERFEJSU

Technické vybavenie bolo navrhnuté
na dostupnej súčiastkovej základni na
báze mikroprocesora MHB 8080 a jeho
podporných obvodov. Aby bolo možné
jednoducho zameniť centrálny proces-
sor za iný, napr. Z80 resp. i8085, resp.
využiť univerzálnu procesorovú časť aj
na riadenie podobného interfejsu pre

COMMODORE C 64, je technické vyba-
venie rozdelené na dve časti:

- a) Centrálny procesor
- b) Interfejs tlačiarne a sériový interfejs
ATARI

Použitie pamätí EPROM 2708 a RWM
2114 bolo vynútené ich relatívne naj-
lepšou dostupnosťou. Ďaleko ekono-
mickejšie je možné postaviť riadiaci
procesor na báze i8085, prípadne MHB
8748. Zapojenie využíva pre vstupno
výstupné operácie adresy z pamäťo-
vého priestoru a adresný dekodér
rozdeľuje pamäť nasledovne:

| | |
|------|-------------|
| 0000 | EPROM |
| 0800 | RWM |
| 1000 | CS1 8251 |
| 1800 | CS2 8253 |
| 2000 | CS3 8255 |
| 2800 | |

Pri adresách vyšších ako 4000H sa
aktivujú tzv. "zrkadlové" adresné prie-
story. Všetky riadiace signály, potrebné
na ovládanie interfejsu sú vyvedené na
30 kontaktovej konektore FRB a plošný
spoj je riešený tak, aby sa signály na
oboch plošných spojoch dostali na
odpovedajúce si kontakty, čo uľahčuje
ich spojenie.

Doska interfejsu obsahuje progra-
movateľný čítač 8253, aby bolo možné
jednoducho vydeliť frekvenciu

19 200x16 pre USART 8251 z Φ 2TTL. Použitie tohto obvodu je možné oceniť, ak je potrebné pripojiť ku adaptéru sériovú tlačiareň a riadiaci program musí prepínať prenosovú rýchlosť z 19 200 na 200, resp. 300 baudov. Pre účely napojenia sériových tlačiarň je technické vybavenie interfejsu doplnené jednoduchým prepínačom na hradlách 7400 s využitím programovo ovládaných výstupov 8251 RTS a CTS. Uvedený riadiaci program nerieši pripojenie sériovej tlačiarne, a v prípade potreby ho je možné modifikovať.

Ovládanie paralelných tlačiarň s interfejsom IRPR/CENTRONICS apod. riadiaci program zabezpečuje prostredníctvom programovateľného obvodu MHB8255. Technické riešenie poskytuje 8 dátových vodičov, 4 riadiace a 4 stavové vodiče. Ošáva ešte nepoužitý port B.

Riadiaci program, uvedený v prílohe, je písaný pre interfejs IRPR, preto sú signály označené názvami tohto interfejsu. Signál DAT/DAT umožňuje jednoduchým nastavením potenciálu log. 0 resp. log. 1 riadiť polaritu výstupných dát. Väčšina tlačiarň umožňuje nastaviť polaritu prepínačom či prepajkou, ale pokiaľ tomu tak nie je, napr. CONSUL 2111, je potrebné meniť riadiaci program, čo spôsobí problémy ak je už adaptér vyrobený a oživený pre iný typ tlačiarne. Podobne v prípade, že vznikne problém polarizácie riadiaceho signálu, sú na plošnom spoji interfejsu dve rezervné pozície pre ľubovoľný 14 resp. 16 vývodový integrovaný obvod so štandardným napájaním, čo by malo stačiť v prípade jednoduchých úprav v zapojení interfejsu.

Podrobnejší popis technického vybavenia nie je potrebný, nakoľko ide o známe a často publikované zapojenie štandardných súčiastok.

POPIS RIADIACEHO PROGRAMU

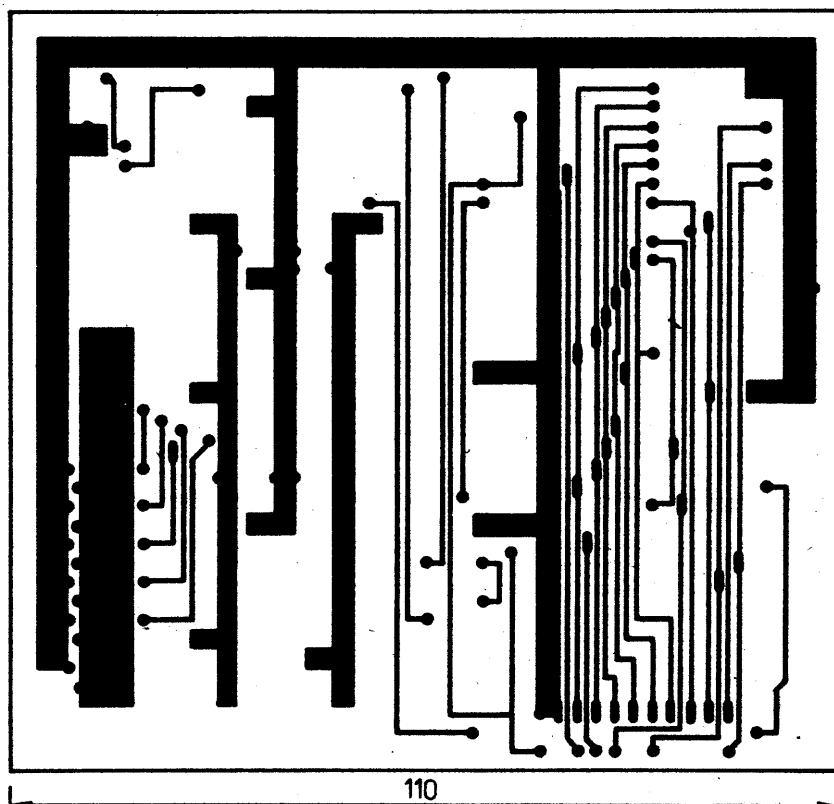
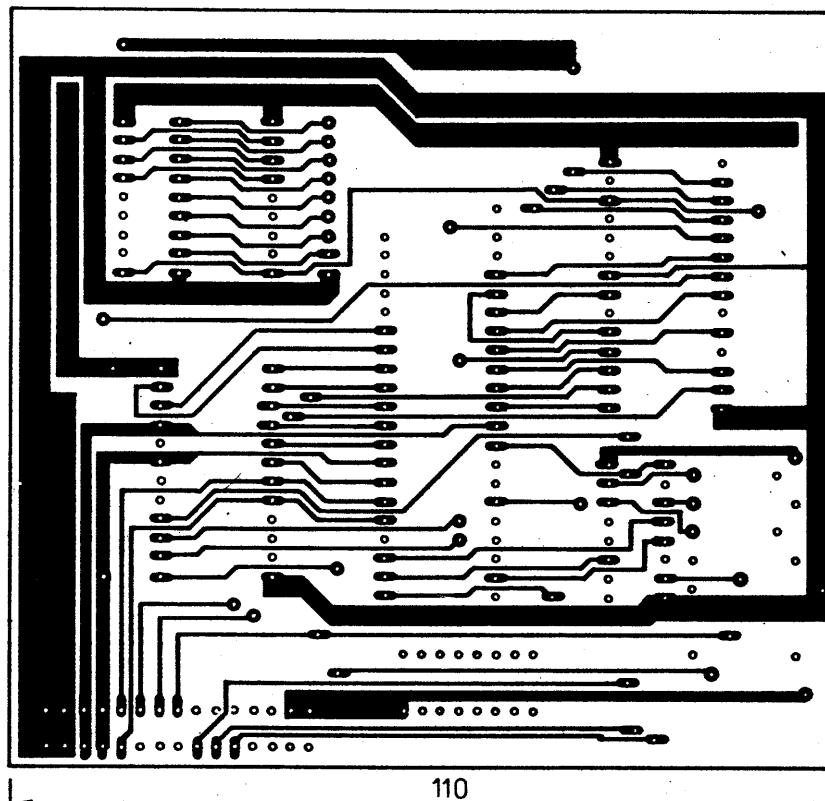
Riadiaci program je možné logicky rozdeliť na dve časti podľa typu použitej tlačiarne:

- a) technicky nezávislú,
- b) technicky závislú.

Nezávislú časť riadiaceho programu tvorí inicializácia programovateľných obvodov a emulovanie ATARI interfejsu pomocou MHB8251.

Podľa typu použitého interfejsu je potrebné modifikovať podprogram TLCZNK — tlač jeden znak na tlačiareň. Tento riadiaci program vyhovuje väčšine tlačiarň dostupných pre užívateľov ATARI u nás. V prípade potreby je v pamäťovom priestore EPROM dostatok voľného miesta, aby bolo možné programovými prostriedkami riadiť aj iný typ zložitejších interfejsov.

```
*****
; Riadiaci program pre atari interface
; Kas - '87
; *****
;
; priradenie
;
; CWTM EQU 1803H ; adr.cw 8253
; TIMO EQU 1800H ; adr.pre TIMO
; CUSART EQU 1001H ; CW 8251
; DUSART EQU 1000H ; dat.reg.8251
; CWP10 EQU 2003H ; CW 8255
; PA EQU 2000H ; adr. port A
```



Obr. 4. Obrázce plošných spojov dosky centrálného procesora X601

```
PB EQU 2001H ; adr. port B
PC EQU 2002H ; adr. port C
PCI EQU 2003H ; bit.oper.PC
```

```
; .phase 0
; LXI SP,0B00H; zasobnik
```

```
; inicializacia obvodov ; 8251
```

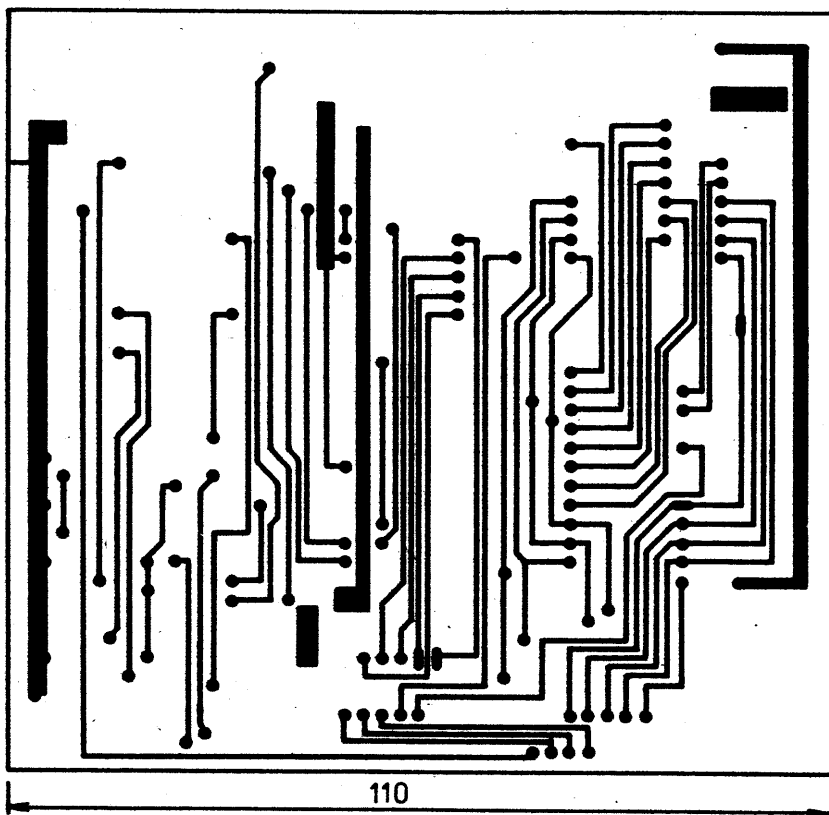
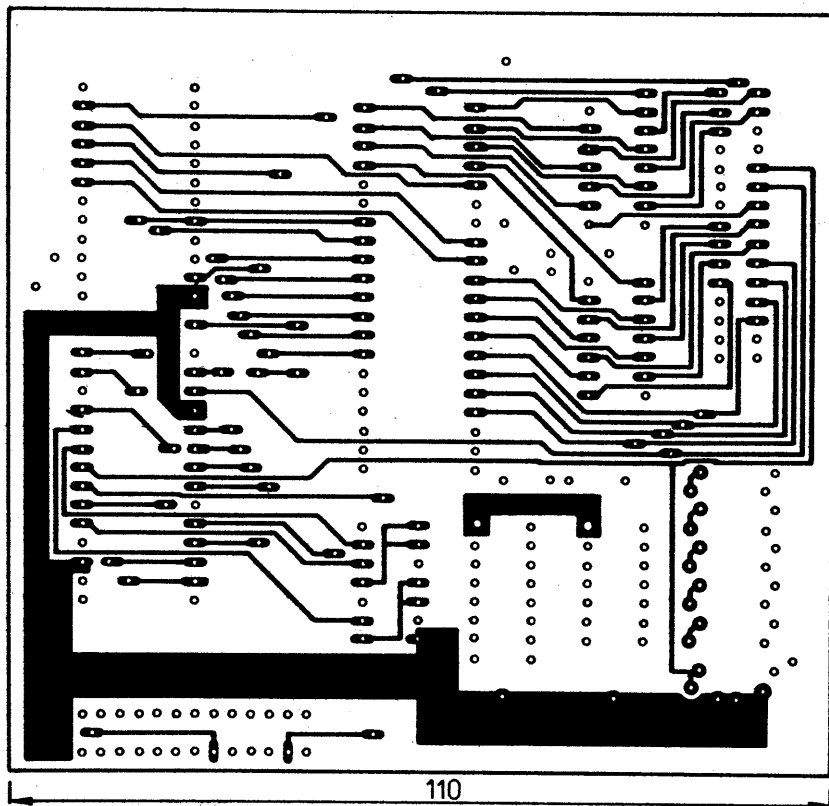
```
; 8253
```

```
; MVI A,36H ; TIMO/MOD 3
```

```
STA CWTM
LXI H,7 ; 19200x16 hz
MOV A,L
STA TIMO
MOV A,H
STA TIMO ; CLK delene 7
```

```
MVI A,4EH ; 8 b./1 stop no par.
```

```
STA CUSART
MVI A,5 ; Rx / Tx
STA CUSART
```



Obr. 5. Obrázky plošných spojov dosky interfejsu X602

; 8255

```
MVI A.83H : PA-out/PCH-out
STA CWP10 : PB-in /PCL-in
MVI A.0DH : MOD 0
STA PC1 : S0/=0 ready
: set PC6
```

test na A0/=0 tlačiaren ready ?

NLED:

```
LDA PC
ANI 8 : A0 je aktivne?
```

JNZ NLED : tlač.not rdy

```
MVI A.0BH : led /ready/
STA PC1 : PC5=1
```

cakaj na kludovy stav atari
cmd nie je aktivny

START:

```
LDA CUSART
RLC
JC START
```

READY:

```
CALL WCMD : cakaj na CMD
MVI B.5
LXI H.CMDBL : 5 riad.znakov
CALL DATA : preber data
CALL WNCMD : koniec cmd ?
LDA CMDBL
CPI 40H : adr.tlaciarene
JNZ START : ina adresa
```

```
CALL ACK : moja adresa
: potvrdim
LDA CMDBL+1 : operacia
CPI 53H : open ?
JNZ DATBL : nie tak data
```

```
MVI B.6 : ak open
: vysli status
LXI H.ODP : z tab.odpoved
```

LC:

```
MOV C,M
CALL VZNK : znak do atari
INX H
DCR B
JNZ LC
```

JMP START : ukončeny open

: prijem dat. bloku v dlzke 40 znakov
: + kontrolna suma do vyrovn. pamate

DATBL:

```
LXI H.BUFF
MVI B.41 : dlzka bloku
CALL DATA
CALL ACK : potvrdenie prevzatia
```

: vytlač prijate data na tlačiaren

```
LXI H.BUFF
MVI B.40
CALL TLC : vytlaci data
CALL ACK : potvrdi sa ukončenie oper
```

JMP START

: podprogrami

: citaj jeden znak z atari

A=citany znak

CZNK:

```
LDA CUSART
ANI 2 : prisiel znak ?
JZ CZNK
```

```
LDA DUSART
RET
```

: zapis znak do atari

C=znak

VZNK:

```
LDA CUSART
RRC : vysielac rdy ?
JNC VZNK
```

```
MOV A,C
STA DUSART
RET
```

: cakaj na CMD signal

WCMD:

```
LDA CUSART
RLC
RC : CMD je aktivny
ANI 2 : prijimac rdy ?
JZ WCMD
LDA DUSART : uvolni prijim.
JMP WCMD
```

: cakaj na koniec CMD

WNCMD:

```
LDA CUSART
RLC
RNC
JMP WNCMD
```

: citaj datovy blok

HL=adr.vyrovn.pamate
B=pocet znakov

```

DATA:
CALL    CZNK
MOV     M.A
INX     H
DCR     B
RZ
JMP     DATA

:
:      vysli ACK do atari
:
ACK:
MVI     C,41H    : kod ACK
CALL    VZNK
RET

:
: *****
: Tato cast programu je premenliva podla
: typu pripojenej tlačiarene. Zmenou
: podprogramu TLC alebo TLCZN je mozne
: menit interface.
: *****

:      vysli data na tlačiaren
:
:      HL=ADR.vyr.pamate
:      B=pocet znakov

TLC:
MOV     C,M
MOV     A,C
CPI     9BH      : ukoncovaci
:      znak dat
JZ      ETLCL   : koniec dat

CALL    TLCZN    : tlać znak
INX     H
DCR     B
JNZ     TLC

RET

ETLCL:
MVI     C,0AH    : kod pre LF
CALL    TLCZN
MVI     C,0DH    : kod pre CR
CALL    TLCZN
RET

:      tlać jeden znak
:
:      C=znak
:
:      / IRPR interface /

TLCZN:
LDA     PC
ANI     1        : AC=0 / rdy ?
JNZ     TLCZN
LDA     PC      : prepojka
:      polarity dat
ANI     2        : data alebo
:      negovane data
JNZ     TL1
MOV     A,C
CMA        : inverzia dat
JMP     TL2
MOV     A,C
STA     PA      : data na interf
MVI     A,0FH   : PC7-strob dat
SC /

J1:     LDA     PC1
ANI     1        : AC=1
:      potvrd.od tlać.
JZ      J1      : cakať na AC

MVI     A,0EH   : zhod SC strob
STA     PC1

J2:     LDA     PC      : cakať na AC=0
ANI     1        : interface rdy
JNZ     J2
RET

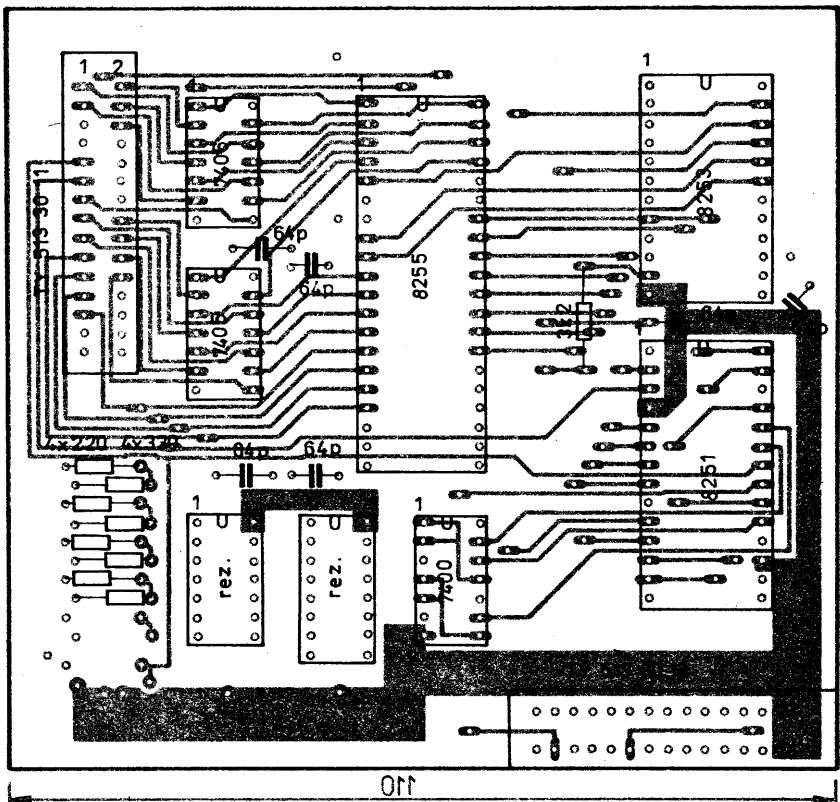
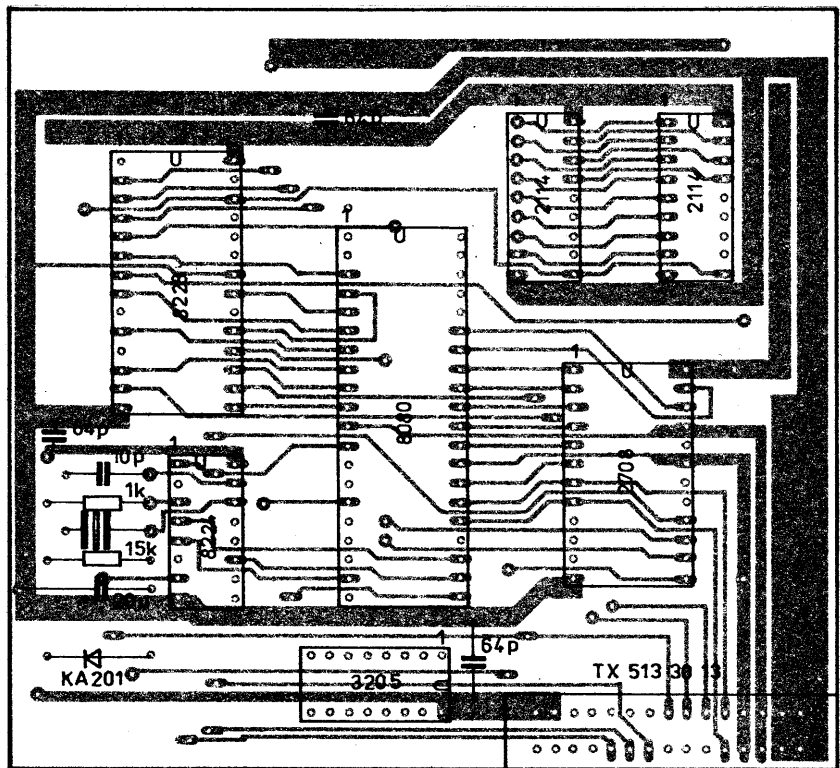
:      tabulky
:
ODP:    DB      43H,80H,4EH,1FH,0,0EDH

:      .dephase
:      org      800h    : RWM priestor.

CMBL:   DS      5      : prikaz.blok
BUFF:   DS      41     : prijatie dat

:
end

```



Obr. 6. Rozloženie súčiastok na doskách procesora a interfejsov X601 a X602 (na doske procesora nutno doplniť rezistor R3 z +5 V na vývod č. 6. IO 3205)

ZÁVER

Zapojenie, riadiaci program, ako aj dosky plošných spojov, boli použité pre stavbu funkčného vzoru, ktorého spolahlivá činnosť bola overovaná na zapojení ATARI 130XE a D100. Tlačovo orientované programy ako SPEED-SCRIPT používali tlačiareň D100 ako ATARI tlačiareň.

Zoznam súčiastok

a) Integrované obvody

| | |
|---------|------|
| MHB8080 | 1 ks |
| MH8228 | 1 ks |
| MH8224 | 1 ks |
| MH3205 | 1 ks |
| MHB2114 | 2 ks |
| MHB2708 | 1 ks |
| MHB8251 | 1 ks |
| MHB8255 | 1 ks |

b) Rezistory

| | |
|------------------|------|
| MH7400 | 1 ks |
| MH7405 | 2 ks |
| KR580VI53 (8253) | 1 ks |
| 1 kΩ | 2 ks |
| 15 kΩ | 1 ks |
| 3,2 kΩ | 1 ks |
| 220 Ω | 4 ks |
| 330 Ω | 4 ks |

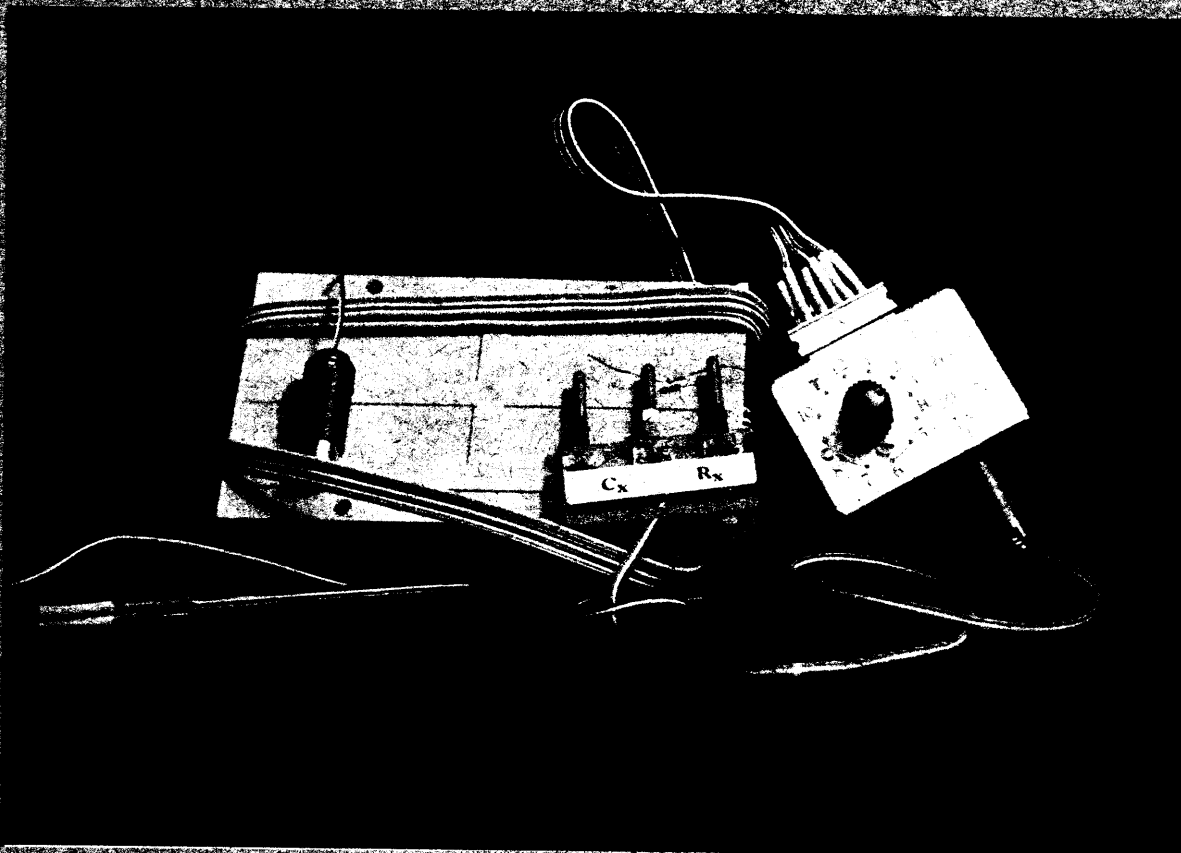
c) Kondenzátory

| | |
|-------------------|---------------|
| 10 pF keramický | 1 ks |
| 64 pF keramický | 9 ks (bypass) |
| 20 μF elektrolyt. | 1 ks |

d) iné

| | |
|--------------------|------|
| dioda KA201 | 1 ks |
| Kryštál 18,432 MHz | 1 ks |

MĚRICÍ PŘÍPRAVEK K A/C PŘEVODNÍKU DIGI 2 PRO IO 151



Přípravek umožňuje měřit vlnu dobrých vlastností převodníku DIGI 2, používajícího k digitálnímu 2. pro měření teploty, kapacity a odporu. Je konstruován jednoduše a leze. Měříte měření dle měřicího převodníku ze předpokladu, že šlo o nebo stavce vlnových techniku má k dispozici počet IO 151 a převodník DIGI 2. Je vhodný zejména tam, kde se s výměrnými přístroji neprovozuje nebo tam, kde je měření teploty, kapacity a odporu.

Ing. Aleš Podroužek, Budovatelů 2743, 407 47 Varnsdorf

Základní technické údaje:

- Měření odporu: 4 rozsahy, od 1 Ω do 30 M Ω
- Měření kapacity: 5 rozsahů, od 10 pF do 80 μ F
- Měření teploty: 1 rozsah, od 0° C do 100° C (v kapalině).

Měření odporu

Pro velikost převodu převodníku DIGI 2 platí podle [1] vztah:

$$P = 65\,535 - 2048 \cdot K \cdot R \quad (1)$$

kde $K = k \cdot C$ ($k=1,16$ bylo zjištěno experimentálně pro dané zapojení MKO 1 podle obr. 2), R je odpor v k Ω a C je kapacita v μ F vnějšího časovacího obvodu. Neznámý odpor rezistoru R_x v k Ω se vypočítá ze vztahu (2), kde C_n je kapacita kondenzátoru (včetně vlastní kapacity přípravku)

včetně vlastní kapacity přípravku) volená přepínačem rozsahů. Sériový rezistor $R_3 = 0,33$ k Ω je na začátku měření kalibrován (proměnná R_9).

$$R_x = (65\,535 - P) / (2375,68 \cdot C_n) - 0,33 \quad (2)$$

Převod P je snímán 10x a do vzorce se dosazuje jeho průměrná hodnota. V tab. 1 jsou informativní údaje pro toto měření — rozsah 6 až 9.

Měření kapacity

Pro hledanou kapacitu kondenzátoru v μ F odvodíme ze vzorce (1), že

$$C_x = (65\,535 - P) / (2375,68 \cdot R_n) \quad (3)$$

kde R_n je celkový odpor rezistoru (včetně sériového odporu R_3 , který je i zde kalibrován). Vlastní kapacita přípravku je přibližně 200 pF. Informativní údaje pro toto měření jsou v tab. 1, rozsah 1

až 5. Teoreticky možnou rozlišovací schopnost udává poslední sloupec tabulky.

Tab. 1. Měření kapacity a odporu

| rozsah | R_n [k Ω] | měřená kapacita C_n [nF] | pF/. převodu |
|--------|---------------------|--|-----------------------|
| 1 | 470 | 0,01 až 5,9 | 1 |
| 2 | 100 | 5,9 až 275 | 4 |
| 3 | 10 | 275 až 2670 | 42 |
| 4 | 1 | 2670 až 21.10 ³ | 421 |
| 5 | 0,1 | 21.10 ³ až 84.10 ³ | 4209 |
| | C_n [nF] | měřený odpor R_n [Ω] | k Ω /. převodu |
| 6 | 10 ³ | 1 až 27 | 0,4 |
| 7 | 100 | 27 až 276 | 4,2 |
| 8 | 10 | 276 až 2760 | 42,0 |
| 9 | 1 | 2760 až 27600 | 420,0 |

Měření teploty

Pro měření teploty byl jako snímač zvolen na trhu dostupný perličkový termistor 13NR15 (odpor $R_{25} = 10 \Omega$ až $30 \text{ k}\Omega$ materiálová konstanta $B = 2800$ až 3600 K , barevné označení — černá a modrá tečka, výkonová citlivost neudána). Pro převodník DIGI 2 by byl vhodnější termistor 14NR15 s větším odporem R_{25} (30Ω až $100 \text{ k}\Omega$) [2]. Závislost odporu termistoru na teplotě θ je dána vztahem

$$R = R_{25} \cdot \exp B \cdot (1/298 - 1/(273 + \theta)) \quad (4)$$

Měříme tedy opět odpor. Z obr. 1 je patrné připojení termistoru k přípravku. V sérii s termistorem je rezistor R_6 , kterým zmenšujeme jeho výkonovou ztrátu a zároveň optimalizujeme velikost převodu P v součinnosti s časovacím kondenzátorem C_1 převodníku DIGI 2. Ten je připojen sekci přepínače rozsahů $Pf1c$ v poloze 11. Výkonová ztráta je menší než 3 mW , tj. méně než 10% z maximálního přípustného zatížení termistoru této řady. Pro měření teploty v kapalině to postačuje pro dosažení termoelektrického režimu. Nelineární závislost odporu termistorů na teplotě je aproximována logaritmickou regresí:

$$\theta = a + b \cdot \ln P \quad (5)$$

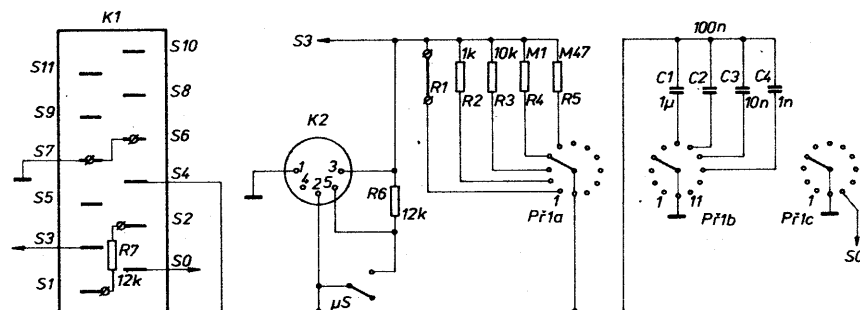
kde a , b jsou součinitele získané regresí. Pracovní rozsah 0 až 100°C byl rozdělen do 6-ti podrozsahů tak, aby korelační součinitel byl přibližně roven jedné (0,999).

Konstrukce přípravku

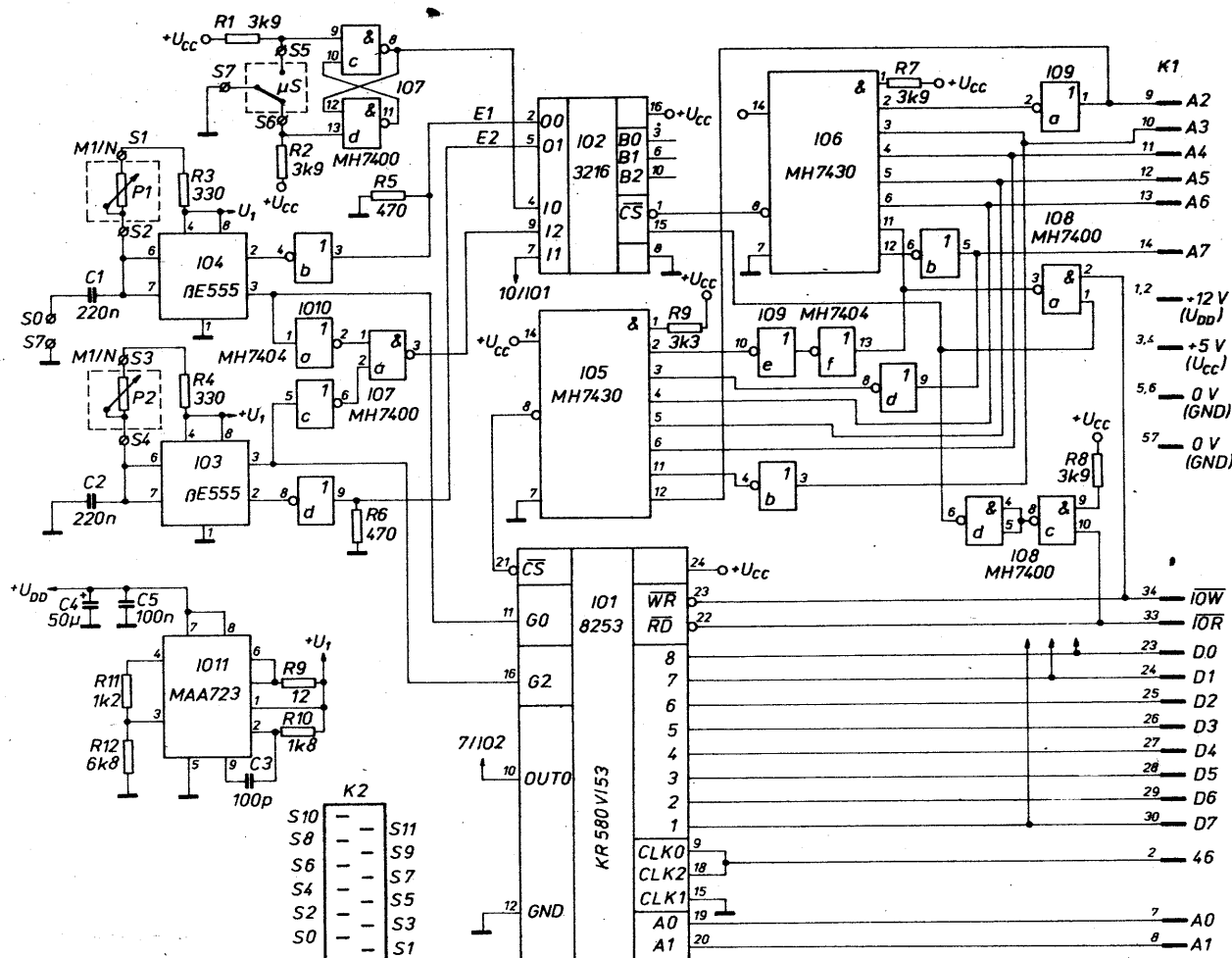
Schéma zapojení je na obr. 1. Vzhled přípravku (i s převodníkem DIGI 2) je na str. 47. Základem je krabička vyrobená z oboustranného kuprexitu tl. 1 mm o rozměrech $70 \times 50 \times 32 \text{ mm}$, spojená pájením ve styčných plochách. Dvanáctipolový třípatrový přepínač (typ APM) je přišroubován v levé polovině krabičky a drží i krycí organické sklo tl. 2 mm , chránící popisový štítek.

Většina součástek je připájena přímo na kontakty přepínače nebo konektorů. Na přední stěně je zásuvka konektoru DIN 5/180. Je připájena po obvodu přírby ke stěně krabičky. Na zadní stěně je přilepena konektorová zásuvka WK46515. Ve volném prostoru vpravo nahoře je připevněn mikrosřpnáč B593, který se používá ke kalibraci sériového rezistoru R_6 při měření teploty.

Pro měření odporu a kapacity se používá držák měřených součástek, který je k přípravku připojen třížilovým plochým ohebným vodičem, zakončeným zástrčkou konektoru DIN. Měřené součástky se upínají do upravených krokosvorek (rozteč 20 mm). Vodiče



Obr. 1. Schéma přípravku na měření R , C , θ



Obr. 2. Schéma zapojení převodníku DIGI2

jsou s krokosvorkami spojeny pájením. Horní krycí destička je též z organického skla o tl. 5 mm.

Před montáží je třeba součástky přesně změřit (R_n , C_n). Usnadní se tím práce při kalibraci přípravku. Rezistor R7 nahrazuje chybějící potenciometr P2 (obr. 2). Rezistor R1 je drátová spojka. Spojení S6 a S7 na konektoru WK nahrazuje chybějící mikrospínač z hledáčku digitizéru. Před připojením přípravku k DIGI 2 zkontrolujeme zapojení konektoru WK. Musí souhlasit s protikusem. Jeho části nejsou zajištěny proti pootočení. Protože otočený konektor nebo nepřipojený přípravek nemohou způsobit havárii ani programu ani převodníku, postačí si pamatovat, že nápisy TESLA na konektoru patří např. k sobě.

Programové vybavení

Program je dostatečně vybaven komentářem. Má dvě části: BASIC a strojový kód, ve kterém je podprogram pro obsluhu programovatelného čítače převodníku DIGI 2. Program je dialogem mezi uživatelem a počítačem. Při správném plnění pokynů, pokud ovšem není přepínač rozsahů v jiné poloze než je zadáno, dojde ke správnému měření. Po přečtení krátkého komentáře se objeví MENU, umožňující další volbu. Při měření odporu a kapacity je nutné po kalibraci zadat rozsah. Při jeho chybném zadání, chybné kalibraci nebo při stisku M, skočí běh programu opět na MENU. Při měření teploty se rozsah volí automaticky. I zde se provádí kalibrace, po které je možné zadat korekci teploty, aby bylo dosaženo souhlasu s teploměrem.

Při ožiování přípravku je třeba změnit hodnoty C_n od č. ř. 1428 a R_n od č. ř. 1528 podle výsledků kalibrace. Podobně je třeba upravit regresní funkce (podprogram č. ř. 8000) a hranice podrozsahů v podmínkách od č. ř. 1080.

Přetečení obsahu programového čítače 8253 při měření odporu a kapacity je hlídáno testem jeho výstupu OUT 0 (čítač Č0). Při vynulování jeho obsahu se zde objeví log. 1, která je zjištěna porovnáním s maskou 7FH. V případě shody pokračuje program vynulováním registrového páru HL a tato nulová hodnota se vrací do BASICu. To je zaregistrováno v podprogramu „rozsah“ a na obrazovce se objeví pokyn „Zvol větší rozsah“. Naopak, pokud je

převod velký (čítač je v činnosti krátkou dobu a rozdíl 65 535 — P je malý), je hlášeno „Zvol menší rozsah“, takže prakticky nelze měřit špatně.

Kalibrace a oživení přípravku

Jak již bylo dříve uvedeno, je pro urychlení seřízení přípravku vhodné přesně změřit R_n a C_n . Odpor sériových rezistorů se zjišťuje programově při každém měření. Kalibrace rezistoru R3 převodníku se provede zkratováním svorek R_n . Jeho odpor se vypočítá ze vzorce (2) bez odečtení hodnoty 0,33 k Ω (tj. přibližný odpor kalibrovaného rezistoru). Podobně se vyhledá i skutečný odpor kombinace R3 + R6 při měření teplot. K doladění přípravku na všech rozsazích byly použity normály. Jejich hodnoty jsou voleny tak, že se měření překrývá na jednotlivých rozsazích. Tím je zajištěno i vzájemné porovnání jednotlivých sousedních rozsahů.

Výpis programu „Měření R, C, θ “ s DIGI2

```

0 CLS
4 PRINT "MERICI PRIPRAVEK"
8 PRINT:GOSUB2600:REM LEGENDA
10 GOSUB2900:CLS:GOSUB2630
12 GOSUB2900:CLS:GOSUB2650
14 GOSUB2900:CLS:GOSUB2700
16 RA=24:GOSUB3000:GOSUB2100:REMP00TRH:TEXT
20 GOSUB2000:REM Cekej
180 DIMQ(20)
182 DEFNZ(X,D)=INT(X*10*D+.5)/10*D
194 REM*****
196 REM MENU
198 REM*****
200 CLS:PRINT&5,0;"M E N U"
210 PRINT&10,0;"1-Mereni odporu"
220 PRINT&2-Mereni kapacity"
230 PRINT&3-Mereni teploty"
240 PRINT&4-Konec programu"
245 PRINT&5-Nahrani programu"
250 GOSUB2050:REM VOLBA
300 UNVAL(V$)GOSUB1400,1500,1000,9999,49990
310 GOTO200
1000 REM*****
1002 REM MERENI TEPLoty
1004 REM*****
1006 CLS:PRINT"MERENI TEPLoty"
1008 PRINT"Pouzity snimac - 13NR15."
1010 PRINT"Rozsah teploty - 0 AZ 100 st.C v kapaline."
1012 PRINT"K převodníku DIGI2 pripoj pri-pravek na mereni teploty."
1014 PRINT:PRINT"Nyni provedeme kalibraci prevodu"
1022 PRINT"Termistor zkratujeme mikrospici cem, který podrzieme a soucas -"
1024 PRINT"ne stiskeme tlacitko."GOSUB2000
1026 PRINT&12,0SPC(255)SPC(255)
1028 GOSUB7000:REM KALIBRACE
1030 RA=24:GOSUB3000:GOSUB2110:GOSUB2000
1034 PRINT&12,0SPC(255);SPC(255);
1036 RA=14:GOSUB3000:REM PODTRZENI
1038 RA=24:GOSUB3000:AP=0:T1=0
1044 GOSUB5000:REM MERENI
1048 R1=HEX(FFFF)-Q-R:T=0
1050 REM PRINT&23,0;"R1="FNZ(R1,0)&23,10;"P^Q="FNZ(Q,0)"R="FNZ(R,0)
1052 IFR1(0 OR R1)&5535-RTHENPRINT&27,0;"Mimo rozsah -> MENU":GOSUB2000:GOTO200
1070 IF A$="M" THEN200
1078 REM 0 az 14
1080 IFR1(37000 AND R1<=59400)THEN 8010
1082 REM 14 az 30
1084 IFR1(18000 AND R1<=37000)THEN 8020
1086 REM 30 az 40
1088 IFR1(11500 AND R1<=18000)THEN8030
1090 REM 40 az 50
1092 IFR1(7900 AND R1<=11500)THEN8040
1094 REM 50 az 70
1096 IFR1(4000 AND R1<=7900)THEN 8050
1098 REM 70 az 100
1100 IFR1(1532 AND R1<=4000)THEN 8060
1106 GOSUB2200:REM TEXT1
1107 PRINT&12,0R0$;SPC(5)
1108 PRINT&10,0;"Merena teplota":PRINT
1109 PRINT"TAB(9) "stupnu Celsia"
1112 T=T+T1:T=FNZ(T,1)
1114 PRINT&20,2T
1116 WAIT(20):PRINT&20,2SPC(7)
1118 IF AP=1 THEN1210
1200 AP=1:PRINT&20,2T&26,0;
1202 INPUT"Korekce teploty";T1:PRINT&26,21"st.Celsia"
1210 IFINKEY$="" THEN1044
1212 GOTO200:REM MENU
1400 REM*****
1402 REM MERENI ODPORU
1404 REM*****
1406 CLS:PRINT&2,0;"MERENI ODPORU"
1408 PRINT"K pravodniku DIGI2 pripoj pri-pravek na mereni odporu a kapa -city
1409 PRINT"Meri odpor v intervalu 1 Ohm - 30 MOhm"
1410 PRINT"Po pripojeni pokracuj stiskem tlacitka "
1412 GOSUB2000:GOSUB7100:REM KALIBRACE R & C
1414 PRINT&10,0SPC(255);SPC(255);SPC(128)
1416 RA=13:GOSUB3000:RA=24:GOSUB3000:GOSUB2100:GOSUB2000
1418 IFA$="4"ORA$="M"THEN200
1420 RA=17:GOSUB3000:IFINKEY$="M"THEN200
1422 PRINT&15,0;:INPUT"Zvol rozsah (6-9)";B$
1423 IFB$("6" OR B$)"9"THEN200
1424 GOSUB5000:GOSUB6000
1426 R1=HEX(FFFF)-Q:REM PRINT&23,0;"R1="R1&23,14"Prevod="Q
1428 IFB$="6"THENC=-.04:D=2
1430 IFB$="7"THENC=-.1:D=1
1432 IFB$="8"THENC=-.945E-2:D=1
1434 IFB$="9"THENC=-1.03E-3:D=0
1436 REMIFB$="10"THENC=3E-3
1438 GOSUB2100:GOSUB2200
1450 R1=R1/2375.68/C-R9
1452 PRINT&19,0;"Mereny odpor"
1454 IF R1=1 AND R1(1000)THEN1480
1456 IF R1(1000)THEN1472
1458 R1=FNZ(R1,4)*1000
1460 IF R1(0 THENR1=0
1464 GOSUB4000:REM ohm
1466 GOTO1482
1472 R1=R1/1000:R1=FNZ(R1,2)
1474 GOSUB4100:REM megaohm
1476 GOTO1482
1480 PRINT&22,0;"R="&22,10"kiloohmu";:R1=FNZ(R1,D)
1482 PRINT&22,2R1;
1490 GOSUB2000:PRINT&22,2SPC(30);
1492 IFA$="M"THEN200
1494 GOTO1418
1498 GOSUB2100:GOSUB2300:GOSUB2000:PRINT&21,2SPC(30);:GOTO1418
1500 REM *****
1502 REM MERENI KAPACITY
1504 REM *****
1506 CLS:PRINT&2,0;"MERENI KAPACITY"
1508 PRINT"K převodníku DIGI2 pripoj pri-pravek na mereni odporu a kapa -city
1509 PRINT"Meri kapacitu v intervalu 10pF - 80 mikrofarad"
1510 PRINT"Po pripojeni pokracuj stiskem tlacitka "
1512 GOSUB2000:GOSUB7100:REM KALIBRACE R & C
1514 PRINT&10,0SPC(255);SPC(255);SPC(128)
1516 RA=13:GOSUB3000:RA=24:GOSUB3000:GOSUB2100

```

```

1518 GOSUB2000:REM CEKEJ
1520 IF A$="4" THEN 200
1522 RA=17:GOSUB3000
1524 PRINT&15,0;:INPUT "Zvol rozsah (1-5)";C$
1528 IFC$="1" OR C$="5" THEN 200
1530 GOSUB5000:GOSUB4000
1532 GOSUB2100:GOSUB2200
1534 R1=HEX(FFFF)-Q
1536 REM PRINT&23,0;"R1="R1;&23,13"Prevod Q="Q
1538 IFC$="1" THEN R1=1026:D=2
1540 IFC$="2" THEN R1=103.99:D=2
1542 IFC$="3" THEN R1=10.1:D=1
1544 IFC$="4" THEN R1=.96:D=0
1546 IFC$="5" THEN R1=0:D=0
1548 C1=R1/2.37568/(R+R9)-.2
1550 PRINT&19,0;"Merena kapacita"
1552 IFC1)=1 AND C1<1000 THEN 1570
1554 IFC1)1000 THEN 1564
1556 C1=FNZ(C1,4)*1000
1558 GOSUB4200
1560 GOTO1572
1564 C1=C1/1000:C1=FNZ(C1,2)
1566 GOSUB4300
1568 GOTO1572
1570 PRINT&22,0;"C="&22,11"nanofarad":C1=FNZ(C1,D)
1572 PRINT&22,2C1
1574 GOSUB2000:PRINT&22,2SPC(30)
1576 IF INKEY$="M" THEN 200
1578 GOTO1526
2000 REM*****
2002 REM CEKEJ
2004 REM*****
2010 A$=INKEY$:IFA$="" THEN 2010
2015 WAIT(10):RETURN
2040 RETURN
2050 REM*****
2052 REM VOLBA
2054 REM*****
2060 V$=INKEY$:IF V$="" THEN 2060
2065 WAIT(10):RETURN
2100 REM*****
2102 REM TEXT
2104 REM*****
2110 PRINT&27,0;"Pokracuj stiskem tlacitka"
2115 RETURN
2200 REM*****
2202 REM TEXT1
2204 REM*****
2210 PRINT&29,0;"MENU - prosim, podrz M";
2215 RETURN
2300 REM*****
2302 REM TEXT2
2304 REM*****
2310 PRINT&29,0;"Navrat na MENU - spatnou volbou
2315 RETURN
2400 REM*****
2402 REM TEXT 3
2404 REM*****
2406 PRINT&25,0;"Zvol vetsi rozsah":GOSUB2100:GOSUB2000
2408 PRINT&25,0SPC(20)
2410 RETURN
2500 REM*****
2502 REM TEXT 4
2504 REM*****
2506 PRINT&25,0;"Zvol mensi rozsah":GOSUB2100:GOSUB2000
2508 PRINT&25,0SPC(20)
2510 RETURN
2600 REM*****
2602 REM TEXT 5
2604 REM*****
2606 PRINT"Legenda"
2608 PRINT"Tento program slouzi pro obsluhu"
2610 PRINT"mericiho pripravku k A/D prevod-"
2612 PRINT"niku DIGI 2. Pripravek umoznuje"
2614 PRINT"mereni odporu rezistoru, kapaci-"
2616 PRINT"ty kondenzatoru a teploty"
2618 PRINT"Spojeni s prevodnikem je prove-"
2620 PRINT"dno konektorem WK napisy TESLA"
2622 RETURN
2630 PRINT"k sobe. Merena soucaska se up-"
2632 PRINT"ne do prislusnych krokosvorek"
2634 PRINT"Rx nebo Cx spojenych konektorem"
2636 PRINT"DIN 5/180.Upnuti soucasky musi"
2638 PRINT"byt spolehlive.Pro mereni teplo-"
2640 PRINT"ty je snimac pripojen tez konek-"
2642 PRINT"torem DIN 5/180. Nasledujici ta-"
2644 PRINT"bulka ukazuje funkce prepinace."
2646 RETURN
2650 REM*****
2652 REM TABULKA
2654 REM*****
2656 PRINT"PREPINAC ROZSAHU"
2658 PRINT&2,1CHR$(15)"P"
2660 M=2:GOSUB2800
2662 PRINT&2,31CHR$(15)"_";&6,1CHR$(15)"S"
2664 M=6:GOSUB2800
2666 PRINT&4,31CHR$(15)"N";&4,1CHR$(15)"K";M=24:GOSUB2800
2668 PRINT&4,31CHR$(15)"J"
2670 S=1:GOSUB2820:GOSUB2850
2672 S=9:GOSUB2820:GOSUB2850
2674 S=20:GOSUB2820:GOSUB2850
2676 S=31:GOSUB2820:GOSUB2850
2678 PRINT&4,2"Poloha"&4,12"Funkce"&3,22"Rozsah"&5,21"od
2680 PRINT&8,2"1 az 5"&7,11"mereni"&9,11"kapacity"
2682 PRINT&8,21"10pF 80mF"
2684 PRINT&13,2"6 az 9"&12,11"mereni"&14,11"odporu"
2686 PRINT&13,21"10hm 30MO";&14,29"-";
2688 PRINT&18,2"11 "&17,11"mereni"&19,11"teploty"
2690 PRINT&18,21"0-100st.C"
2692 PRINT&22,2"12 "&22,13"zem"&22,23CHR$(15)"QQQQQ"
2698 RETURN
2700 REM*****
2702 REM TEXT 6
2704 REM*****

```

```

2706 PRINT"Pri mereni dane veliciny je nut-"
2708 PRINT"ne zvolit rozsah dle tabulky"
2710 PRINT"a ridit se instrukcemi."
2712 PRINT"Kdyz presto nedojde k mereni,po-"
2714 PRINT"citac nereguluje na stisk tlacit-"
2716 PRINT"ka, je chyba v pripevneni sou-"
2718 PRINT"castky nebo je soucaska vadna."
2720 PRINT"Pri mereni velkych hodnot soucas-"
2722 PRINT"tek je pri nevhodnem rozsahu do-"
2724 PRINT"ba mereni delsi.Zmenste rozsah."
2750 RETURN
2800 REM*****
2802 REM LINKA vodorovne
2804 REM*****
2808 FOR I=27030:PRINT&N,1CHR$(15)"Q":NEXT
2810 RETURN
2820 REM*****
2822 REM LINKA svisle 1
2824 REM*****
2828 FOR I=3705:PRINT&I,SCHR$(15)"T":NEXT
2830 RETURN
2850 REM*****
2852 REM LINKA svisle 2
2854 REM*****
2858 FOR I=77023:PRINT&I,SCHR$(15)"T":NEXT
2860 RETURN
2900 REM*****
2902 REM STRANKA
2904 REM*****
2906 RA=26:GOSUB3000:GOSUB2100:GOSUB2000:RETURN
3000 REM*****
3002 REM PODTRH
3004 REM*****
3010 FOR I=07030:PRINT&A,1CHR$(15)"Q":NEXT:RETURN
4000 REM*****
4002 REM TISK VYSLEDKU-ODPOR 1
4004 REM*****
4006 PRINT&22,0;"R="&22,11"ohm";
4008 RETURN
4100 REM*****
4102 REM TISK VYSLEDKU-ODPOR 2
4104 REM*****
4106 PRINT&22,0;"R="&22,11"megohm";
4108 RETURN
4200 REM*****
4202 REM TISK VYSLEDKU-KAPACITA1
4204 REM*****
4206 PRINT&22,0;"C="&22,11"pikofarad";
4208 RETURN
4300 REM*****
4302 REM TISK VYSLEDKU-KAPACITA2
4304 REM*****
4306 PRINT&22,0;"C="&22,11"mikrofarad";
4308 RETURN
5000 REM*****
5002 REM MERENI
5004 REM*****
5006 REM MERENI
5008 FOR I=07020
5010 Q(I)=0
5012 NEXT I
5014 FOR I=07020
5016 Q(I)=WORD(HEX(4000),1)
5018 NEXT I
5050 Q=0
5055 FOR I=17020
5060 Q=Q+Q(I)
5062 REM PRINTQ;Q(I)
5065 NEXT I
5070 Q=FNZ(Q,1)/20
5080 RETURN
6000 REM*****
6005 REM ROZSAH
6010 REM*****
6015 FOR I=07010
6020 IF Q(I)<1000 THEN 6100
6030 IF Q(I)>65100 THEN 6200
6040 NEXT I
6050 RETURN
6100 REM*****
6105 REM Maly rozsah
6110 REM*****
6112 GOSUB2400
6114 IF V$="1" THEN 1410
6118 IF V$="2" THEN 1526
6200 REM*****
6205 REM Velky rozsah
6210 REM*****
6212 GOSUB2500
6214 IF V$="1" THEN 1418
6218 IF V$="2" THEN 1526
7000 REM*****
7002 REM Kalibrace teploty
7004 REM*****
7020 GOSUB5000
7022 R=HEX(FFFF)-Q
7024 PRINT&14,0;"R="R;&14,13"Prevod Q="Q
7050 RETURN
7100 REM*****
7102 REM Kalibrace R & C
7104 RA=14:GOSUB3000
7106 PRINT"Kalibrace mereni R & C"
7108 PRINT"Zvolime rozsah ..... 6"
7110 PRINT"Zkratujeme svorky ... R"
7112 PRINT"Stiskneme tlacitko a tim je ka- librace skoncena"
7114 RA=25:GOSUB3000:GOSUB2000
7130 GOSUB3000
7132 R1=HEX(FFFF)-Q
7134 R9=R1/1995.3712:R9=FNZ(R9,3)
7136 PRINT&27,0;"Odpor prevodniku R="R9"kiloohm";WAIT(25)
7138 RETURN
8000 REM*****
8002 REM PPM TEPLOTA
8004 REM*****
8010 R0$="Rozsah 0 az 14 st.C"
8012 T=219.0097-19.931117*LOG(R1)+.16
8014 GOTO1184
8020 R0$="Rozsah 14 az 30 st.C"
8022 T=292.53206-26.759259*LOG(R1)
8024 GOTO1184

```

```

8030 R0s="Rozsah 30 az 40 st.C"
8032 T=251.96486-22.595167*LOG(R1)
8034 GOTO1186
8040 R0s="Rozsah 40 az 50 st.C"
8042 T=282.75638-25.98703*LOG(R1)
8044 GOTO1186
8050 R0s="Rozsah 50 az 70 st.C"
8052 T=310.19176-28.96891*LOG(R1)
8054 GOTO1186
8060 R0s="Rozsah 70 az 100 st.C"
8062 T=330.28362-31.404993*LOG(R1)
8064 GOTO1186
9999 END
49990 REM*****
49992 REM NAHRANI PROGRAMU
49994 REM*****
49996 PRINT&24,0;"Prosím, zapni magnetofon a stiskni nejaké tlačítko."
49998 IF INKEY$="" THEN 49998
49999 PRINT&20,0;"Nahrávám !"
50000 CALL HEX(6000)
50002 PRINT&24,0;"Prosím, vypni magnetofon a po --"
50004 PRINT&24,0;"kracuj volbou z menu. " &28,0SPC(10);
50006 GOSUB 2000:CLS:RUM180

```

Výpis podprogramu „Měření časové prodlevy s 8253“

```

4000 F3 DT ; začneme všechna přerušeni
4001 3E 30 MOV A 30 0 ; inicializujeme čítač. Číslo - řídicí slovo do stádače
4003 D3 77 OUT 77 w ; a odtud do čítače
4005 3E FF MOV A FF 0 ; slabika FFH obsahu čítače do stádače
4007 D3 74 OUT 74 t ; naplnění nižšího bajtu
4009 D3 74 OUT 74 t ; vyššího bajtu čítače
400B 79 MOV A,C ; adresa inicializovaného MKO z reg.C do stádače
400C D3 79 OUT 79 y ; nahození MKO
400E D3 79 IN 79 y ; je MKO ještě nahozen? Jestliže ano, čtu log 0 na D7.
4010 F7 ORA 4 ;
4011 FA 0F 43 JM 400E ; a vracím se opět na čtení D7, dokud nepřijde log 1
4014 FE 7F CP 7F 0 ; pokud se nečítalo znovu, tak budu číst obsah čítače
4016 FA 21 40 JM 4001 ; jinak jdu na nulování reg. páru H,L
4019 DB 74 IN 74 t ; čtení nižšího bajtu
401B 67 MOV L,A ; ukládám ho do registru L
401C DB 74 IN 74 t ; čtení vyššího bajtu
401E 67 MOV H,A ; ukládám ho do registru H
401F F8 EI ; povolují přerušeni
4020 C9 RET ; a vracím se z podprogramu do BASICu
4021 21 00 00 LXI H 0000 ; nulují obsah reg. páru H,L, protože načítaná hodnota je chybná
4024 C9 RET ; a vracím se do BASICu

```

Hodnoty R_n a C_n v programu jsou postupně upraveny. Nejpracnější a časově nejnáročnější je kalibrace termistoru. Problém je i v tom, že se jeho vlastnosti mohou časem změnit. Pokud je k dispozici ultratermostat, který cejchovanou teplotu umožňuje udržet dostatečně dlouhou dobu, je výsledek kalibrace velmi dobrý. Je třeba dbát i na stejnou hloubku ponoru snímače a teploměru. Náhradou za ultratermostat může být nádrž s možností vytápění o objemu asi 5l, do které vložíme menší nádobu se snímači. Rychlost ohřevu nebo chlazení musí být dostatečně malá.

Přesnost měření

Přesnost měření je závislá, jak již bylo uvedeno, na velikosti převodu. Z tohoto důvodu je ošetřena programově volba rozsahu. Informativně platí, že časové prodlevě 1 ms odpovídá převod 2048 jednotek (relativní chyba asi 0,05 %). V praxi, při měření odporu v porovnání s naměřenými hodnotami číselným multimetrem RFT G 1001.500, to představuje chybu menší než 1 %. Při měření kapacit a odporů se porovnávaly výsledky měření s přístrojem RLC 10. Byly zjištěny rozdíly až 10 %. Jiné měřidlo není momentálně k dispozici. Při měření teploty závisí chyba na kvalitě kalibrace a přesnosti aproximace. Nemusí překročit 0,1 °C. Lze využít malé tepelné kapacity perlickového termistoru (malá setrvačnost) ke sledování různých fyzikálních dějů, např. fázových přeměn tuhoucích vosků apod.

Závěr

Měřicí přípravek byl postaven jako doplněk digitizéru 2 pro další využití jeho převodníku DIG12. Náklady na jeho stavbu jsou asi 250 Kčs. Mate-

riálové náklady na DIG12 jsou asi 700 Kčs. Jednoduchá konstrukce a dobrá reprodukovatelnost výroby umožňují i amatérskou stavbu. Podmínkou použití je počítač IQ 151 se standardním vybavením. Přípravek má jednu polohu přepínače neobsazenou, takže je možné připojit ještě další snímač odporového nebo kapacitního typu.

Literatura

- [1] Podroužek, A.: Digitizér 2. Varnsdorf, SPŠ 1986.
- [2] Termistiry. Ročenka sdělovací techniky. Praha, SNTL 1965.
- [3] Podroužek, A.: Jednoduchý digitizér. Amatérské radio 8, 9/87.
- [4] Mikroprocesor 8080 a základní obvody. ČSVTS 1982.

Konstrukce, stavba a oživení převodníku DIG12 pro IQ 151

Konstrukce převodníku

Konstrukce převodníku DIG12 je odvozena od předchozího typu převodníku určeného pro jednoduchý digitizér [3]. DIG12 je slučitelný s mechanikou tohoto zařízení. Jeho vlastnosti jsou lepší v porovnání s předchozím typem v dosažené reprodukovatelnosti a rozlišovací schopnosti. Ve spojení s vylepšenou mechanikou vznikl Digitizér 2, jehož popis však není předmětem tohoto příspěvku. Dále budou popsány pouze podstatné změny, neboť popis předchozího převodníku byl uveřejněn.

Místo programového čítače byl použit programovatelný čítač KR580VI53

(8253) IO1 (viz též obr. 2), který obsahuje tři nezávisle pracující sestupné čítače. K němu přibyl adresový dekodér IO5, který vytváří signál CS pro výběr čítače. Adresa je 74 až 77H. Dva použité čítače C0 a C2 pracují nezávisle v módu 0, s přerušením na konci čítání. Výstup OUT čítače je po volbě módu na úrovni L. Po vložení nové předvolby OUT zůstává ve stejné úrovni a čítač zahájí čítání po příchodu H na vstup GATE. Po dočítání na nulu přejde OUT na úroveň H a zůstane na ní, dokud registr čítače není opět předvolen. Výběr čítače se provádí adresovými bity A1 a A0 (předvolba čítače C0 — L, L: C1 — L, H: C2 — H, L). Pro zápis řídicího slova (CW) jsou tyto bity H, H. Tvar slova CW zapisovaného do registru CWR je následující:

| bit | D7 D6 | D5 D4 | D3 D2 D1 | D0 |
|--------|--------------|-------------|------------|---------|
| význam | volba čítače | čtení/zápis | volba módu | čítání |
| | | | | binární |
| | | | | deka- |
| | | | | dicke |

V našem případě pro mód 0, binární čítání a zápis/čtení v pořadí nižší bajt — vyšší bajt, jsou tvary řídicích slov pro čítač C0 — 30H, C1 — 70H a C2 — B0H. Obsluha čítače je patrná i z výpisu programu ve strojovém kódu. Adresa pro zápis CW je 77H a pro čtení/zápis (IN/OUT) čítače C0 je 74H. Na vstup CLK jednotlivých čítačů je přiveden hodinový kmitočet 2048 kHz. Spouštěcí impuls pro GATE je u čítače C0 získán z výstupu 3 IO4, zapojeného jako monostabilní klopný obvod (MKO). Jeho vstup 2 je v klidové úrovni H, což zajišťuje IO10b a rezistor R5. MKO je nastaven bitem D0 programově přes IO2, který musí být aktivován signálem CS, vytvářeným adresovým dekodérem IO6, je-li adresován 78 až 7BH. Směr toku dat obousměrného oddělovače datové sběrnice je řízen signálem DC, který je vytvořen dvěma hradly IO8c, d, ve funkci logického součinu H.IOR. Při čtení dat je DC = L. Druhý bit datové sběrnice D1 je určen též pro hlášení přetečení čítače, které je signalizováno úrovní H výstupu OUT 0 v průběhu čítání. Třetí bit D2 je zde ve významu ukončení časové prodlevy, hlášení z výstupu MKO o úrovni L je dáno přes logický součet IO10a a IO7a. Umožňuje další pokračování programu bez zbytečného čekání.

V případě, že převodník bude využit pouze k měřicímu přípravku, lze vypustit IO3, obvod pro ošetření mikrospínače a logický součet IO10a, c a IO7a lze nahradit sledovačem ze dvou invertorů.

Velikost převodu DIG12 je asi 29× větší než předchozího převodníku. Proto došlo i ke zmenšení kapacity časovacího kondenzátoru MKO. Relativní chyba převodu je asi 0,05 %. Velikost převodu je úměrná rozdílu hodnoty předvolby čítače (FFFFH = 65 535 a obsahu čítače po dočítání).

Stavba převodníku

Protože se v amatérských podmínkách předpokládá jednorázová kusová výroba a výroba oboustranných desek plošných spojů je obtížná, bylo použito

technologie ovíjených spojů drátem se samopájatelnou izolací o průměru 0,1 mm. Po rozmístění součástek na desku (může být užito univerzální desky, umožní lepší upevnění součástek) z libovolného izolačního materiálu nepodléhajícího tepelné degradaci o rozměrech 135x92 mm se součástky propojí přímo podle schématu. Ideální k tomu účelu je stavebnice Univerzal, kterou vyrábí ZPA Nový Bor a do škol dodává n. p. Komenium. Součástí této stavebnice je ovíjítko. Pájecí body tvoří zpravidla vývody IO nebo jejich patič. Ovíjítkem vytvoříme v pájecím bodě dva až tři závit, drát utahujeme a pokračujeme k dalšímu spojovanému bodu, kde po ovinutí drát přerušíme pomocným nástrojem, který slouží i pro vedení drátu (držíme jej v levé ruce). Pájení je třeba provádět čistě s minimálně nutným množstvím pájky a přiměřeným množstvím tavidla. Spojové vedeme přímo, pečlivě a pozorně, nesmí dojít k nekontrolovatelnému poškození izolace teplem pájecího hrotu. Spojení převodníku s mechanikou nebo měřicím přípravkem je provedeno ohebným plochým vodičem 8x0,3 nebo 8x0,5, který je veden v drážce 11x2,8 mm na spojovací desce a na druhé straně je zakončen konektorem WK 46240 se 12-ti vývody.

Oživování převodníku

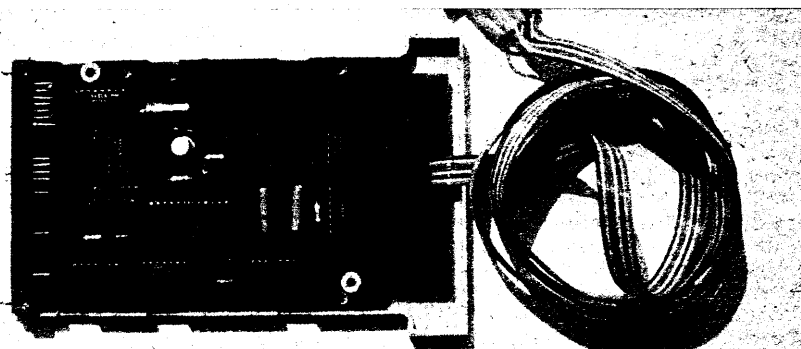
Před oživováním se vyplatí pečlivá kontrola všech spojů. Je to obtížné, ale vyplatí se to. Teprve potom připojíme desku ke zdroji. Vzorek byl oživován laboratorním zdrojem BS525. Kontrolujeme zejména napětí stabilizátoru, které má být asi 5 V. Odběr převodníku by měl být okolo 250 mA (U_{cc}). Aby nevznikly úbytky napětí v napájení IO, je třeba tyto spoje náležitě zesílit (několik průřezů nebo použít tlustší vodič). Do vypnutého počítače zasuneme výnos, do něho převodník (nezakrytovaný), zapneme počítač a monitor a sledujeme jeho obrazovku. Pokud se objeví za přítomnosti modulů BASIC 6 a VIDEO32 známé *Basic — ready*, je vše zatím v pořádku. Pokud se objeví směsice znaků po celé obrazovce, počítač vypneme a odstraníme zkrat na adresové sběrnici. Pokračujeme nahráním programu. Ten se skládá ze tří částí a jeho nevýhodou je, že je příliš dlouhý. V této fázi potřebujeme pouze část z přílohy 2 a k tomu přidáme v BASICu podprogram „Měření“ řádek 5000. Na řádku 5062 zrušíme REM a na řádek 5080 vložíme místo RETURN např. WAIT (50): GO TO 5000. Je účelné si i tento krátký program spolu s programem ve strojovém kódu od adresy 4000 nahrát. Zajistíme se tak proti jeho ztrátě při zacyklení počítače. Pokud se po odstartování programu objevují na obrazovce snímané převody (je pochopitelné nutné mít připojení mechaniku nebo měřicí přípravek — pozor na špatně připojený nebo otočený konektor WK), je vše v pořádku. V případě, že převodník pracuje špatně, kontrolujeme jednotlivé signály, počínaje CS pro IO1 a IO2, spouštěcí impuls pro MKO nebo GATE čítače, stav OUT čítače apod. Ke kontrole stačí školní logická sonda BK 121. Po oživení zabudujeme desku do modulové krabičky, kterou opatříme

Výpis strojového kódu spojovacího programu

```
6000 C3 43 61 C3 64 62 C3 88 63 06 54 05 C2 0B 60 DB CCaCdbC c T B 'L
6010 B6 47 DB B6 A8 F2 12 60 78 07 9F F6 66 D3 86 78 Gf (r 'x v S x
6020 C9 0E 01 CD 09 60 AA 07 79 17 4F 02 23 60 32 FF I M 'y OR# '20
6030 EF 83 5F C9 CD 21 60 61 CD 21 60 69 C9 21 00 00 o _IM! 'ah! 'ii!
6040 22 F8 5F 2E 4C 3E 06 D3 86 CD 12 60 F3 4F CD 09 "x..L) S M 'sGH
6050 60 A9 F2 4E 60 CD 09 60 A9 FA 60 60 CD 09 60 A9 ')*N'M 'z' 'H '
6060 A9 57 1E 00 CD 21 60 AD C2 43 60 CD 34 60 E5 CD 'W M! -BC'M4'eM
6070 34 60 D5 CD 21 60 D1 79 B8 C2 27 F2 83 5F D5 EB 4'UM! 'Qy;B'r _UK
6080 2A F8 5F 19 D1 C1 C5 CD 21 60 71 C1 23 0B 78 B1 o x _QAEM! 'q# 'x1
6090 C2 86 60 E5 CD 34 60 D5 EB 2A F8 5F 19 D1 C1 79 B 'eM4'Uk*x _QAy
60A0 AD C2 92 F7 78 AC C2 92 F7 CD 34 60 D5 CD 21 60 -B wx,B wM4'UM! '
60B0 D1 79 B8 C2 92 F7 C1 FB E9 22 FA 5F EB CD F5 61 Qy;B wA(i'z_kMud
60C0 23 22 FC 5F 60 69 22 FE 5F 3E DF D3 89 3E 06 D3 # 'I 'i' 'v _S ) S
60D0 86 1E 06 0E 00 CD 27 61 10 C2 D5 60 1E FC 0E 04 M'a LM'a#i_M;ae#
60E0 CD 27 61 0E 4C CD 27 61 2A FC 5F CD 38 61 E5 2A z_M;akM'a#z_DM#
60F0 FA 5F CD 38 61 4B CD 27 61 2A FA 5F 4E 23 CD 27 ac+iScB! 'wz_DM#i
6100 61 E3 2B 7C B5 C3 C2 FC 60 2A FA 5F 44 40 2A FC z_M;akM'a#z_M;akM'a#A
6110 5F 09 CD 38 61 2A FE 5F CD 38 61 4B CD 27 61 C1 > S C'u ( _M uy
6120 3E 9F D3 89 C3 A2 F5 16 08 7B 81 5F CD 93 F5 79 > S C'u ( _M uy
6130 07 4F F6 06 D3 86 15 C2 2C 61 C9 4C CD 27 61 40 Ov S B,aILM'aM
6140 C3 27 61 3E 02 32 1C 00 2A CE 00 22 CE 61 21 00 C'a) 2 *N 'Na!
6150 60 11 B8 60 01 3D 60 CD 67 F2 2A 54 62 EB 21 00 'B ')*Mgr#Tbk!(
6160 63 01 3D 60 CD 89 60 21 1F 62 11 63 62 01 1F 62 c =M9'! b cb b
6170 CD 89 60 3E 45 CD A5 F5 21 E9 60 11 1F 62 01 3D M9'EMZu!9' b =
6180 60 CD 89 60 3E 06 CD A5 F5 2A A6 00 EB 2A A4 30 'M9' MZu# k#
6190 01 3D 60 CD 88 CE C4 B9 60 2A A2 61 7C B5 CA AD =M8ND9' *a15J-
61A0 61 21 00 40 11 25 40 01 3D 60 CD 89 60 CD FC 61 a! @ XQ =M9'Mia
61B0 21 A0 00 11 D1 00 01 EB 61 CD 89 60 2A 00 00 EB ! Q haM9' *P k
61C0 2A CE 00 01 D0 61 CD 89 60 3E 8A D3 87 C9 6A 01 *N PaM9' ) S Ij
61D0 3E 8A D3 87 3E 02 32 1A 00 2A 42 00 22 CE 00 CD > S ) 2 *B 'N M
61E0 FC 61 CD 7C CD C3 0A CF 2A CE 61 EB 2A 42 00 CD laMHCZ0#Nak#B M
61F0 F5 61 C3 40 60 7D 93 6F 7C 9A 67 C9 2A CE 00 54 uaC0' ) ol gI*N T
6200 5D 7E 23 B6 CA 18 62 23 23 23 AF BE 23 C2 0B 62 J#6J b#6# /#B b
6210 EB 73 23 72 EB C3 FF 61 EB 23 23 22 D0 00 C9 CD ks#rkCak#6#P IM
6220 2B 62 2A CE 00 22 42 00 C3 3D 60 CD 47 F6 1F 3E +b*N 'B C='MGv >
6230 1F 32 13 00 2A 54 62 EB 21 A8 63 7E FE 40 CA 56 2 *Tbk! (c'v#QJv
6240 62 32 0B 00 CD 47 F6 18 CD 9B F4 D2 3B 62 3E 1E b2 MGv M tr;b)
6250 32 13 00 C9 DA 66 23 46 23 CD 47 F6 20 05 C2 59 2 IZf#F#MGv BY
6260 62 C3 3B 62 FF 00 FF 00 FF 00 FF 00 FF 00 FF bC;b# ♦ ♦ ♦ ♦ ♦
```

štítkem DIGI2 orientovaným stejně jako na ostatních modulech IO 151 (nesmí dojít k opačnému zasunutí do počítače). Nyní je možné vložit do počítače celý program. Spojovací část programu (viz výpis) slouží pro jeho nahrávání. Obsahuje hlavičku programu a teplej start BASICu. Program lze celý nahrát stiskem "5" programového MENU. Do počítače se pak vkládá známým postupem pro nahrávání v monitoru. Převodník DIGI2 je součástí stále se rozšiřující stavebnice. Ve spojení

s vylepšenou mechanikou vznikne Digi-tizér 2. Jeho programové vybavení je dnes již dost rozsáhlé a rovněž se neustále rozšiřuje, takže se nevyplatí program ručně vkládat z výpisu, ale nahrát jej přímo na kazetu nebo disketu. To platí i pro měřicí přípravek. Dokumentace a programové vybavení je k dispozici na SPŠ ve Varnsdorfu, kde vám rádi pomohou radou i skutkem, ovšem pochopitelně v rámci ko-nečných časových možností.



Součástky použité pro stavbu převodníku

Integrované obvody

| | |
|----------|---------------------|
| IO 1 | KR 580 VI 53 (8253) |
| IO 2 | MH 3216 |
| IO 3, 4 | E 555 (NE 555) |
| IO 5, 6 | MH 7430 |
| IO 7, 8 | MH 7400 |
| IO 9, 10 | MH 7404 |
| IO 11 | MAA 723 |

Rezistory TR 151 (161)

| | |
|----------|--------|
| R1, 2 | 3,9 kΩ |
| R3, 4 | 330 Ω |
| R5, 6 | 470 Ω |
| R7, 8, 9 | 3,9 kΩ |
| R10 | 12 Ω |
| R11 | 1,8 kΩ |
| R12 | 6,8 kΩ |

R13

P1, 2 1,2 kΩ 0,1 MΩ/N TP 280

Kondenzátory

| | |
|-------|--|
| C1 | TE 007 100 μF/15 V |
| C2 | TK 747 100 nF |
| C3 | TK 747 100 pF |
| C4, 5 | TC 215 220 nF (MPT — Pr 96, TGL 55163) |

Ostatní součástky

| | |
|---------------------------|------|
| Objímka DIL 14 | 6 ks |
| Objímka DIL 16 | 2 ks |
| Objímka DIL 24 | 1 ks |
| Konektor FRB TY 517 20 11 | 1 ks |
| Konektor WK 462 40 | 1 ks |
| Konektor WK 465 40 | 1 ks |

RIDS

Ing. Jaroslav Švehla, Voroněžské nám. 2, 625 00 Brno

K uchování většího množství dat různého charakteru byl vytvořen relační interaktivní databázový systém — RIDS — pro mikropočítače řady MZ-700 a MZ-800 (v módu MZ-700). Tento systém umožňuje interaktivní formou definovat, modifikovat a zpracovávat data dvou typů v tabulkové formě.

Databáze je organizována ve formě tabulek pojmenovaných vlastním jménem. Řádky tabulky představují jednotlivé záznamy databáze, kde každý záznam má volitelný počet položek (sloupců) libovolného jména a formátu.

Příklad:

Jméno databáze: LIDÉ

| 1 | 2 | 3 | 4 |
|-------|----------|---------|-----|
| JMÉNO | PŘÍJMENÍ | TELEFON | VĚK |
| Jana | Krátká | 3227 | 20 |
| Ivoš | Pech | 4138 | 17 |
| Oto | Mann | 2256 | 31 |

Příklad ukazuje organizaci databáze LIDÉ, která obsahuje tři nezávislé záznamy o čtyřech položkách (JMÉNO, PŘÍJMENÍ, TELEFON, VĚK).

FORMÁT DAT

Databáze rozlišuje dva formáty dat:

formát 1 ... numerická data (real nebo integer),

formát 2 ... textová data (řetězce libovolných znaků včetně české abecedy).

Obsah položky je nulový, jestliže = 0 (formát 1) anebo = „..“ (formát 2).

UKAZATEL ZÁZNAMU

Ve struktuře databáze je obsažen ukazatel, který ukazuje na jeden ze záznamů databáze. Tento záznam se nazývá „běžný záznam“ a je to záznam, se kterým se momentálně pracuje a na kterém (resp. od kterého) se budou provádět příkazy.

Obsah tohoto ukazatele se mění buď v souladu s výsledkem určitého příkazu anebo jej může uživatel změnit přímo.

VÝBĚR ZÁZNAMŮ

Databáze umožňuje výběr libovolné podmnožiny záznamů (příkazy O, A, H — viz „PŘÍKAZY“), podle zvolených kritérií, nad kterými se budou provádět příkazy D, P, X (viz „PŘÍKAZY“). Tyto vybrané záznamy se nazývají „aktivní záznamy“.

IMPLEMENTACE

Uvedený interaktivní databázový systém byl realizován na počítači SHARP MZ-821 (v módu MZ-700) ve dvou modifikacích:

- a) v jazyku S-BASIC (MZ-1Z013) — interpret;
- b) ve strojovém kódu, jako produkt kompilátoru S-BASIC COMP.

Pro lepší názornost uvádím modifikaci v S-BASICu. Uživatel má možnost po nahrazení konstrukcí IF ERROR GOTO a RESUME za BREAK ON a BREAK OFF a jiným postupem definice české abecedy (viz ř. 12—13) uvedenou verzi zkompilovat.

STRUKTURA DAT

Vzhledem k tomu, že program byl vytvářen pro základní konfiguraci mikropočítače, bez vnějších periférií pouze s magnetofonem, byla jediná struktura, dostupná pro uchování dat v paměti, pole. Tato skutečnost způsobuje výrazné degradování databáze v oblasti její variability a rozsahu.

Pro fyzické uchování dat bylo využito dvou polí, které uchovávají jak informace o struktuře databáze, tak vlastní data, která tvoří její obsah.

A. Pole ADF\$(N)

| 0 | 1 | 2 | ... | X | X+1 | X+2 | ... | N |
|----------------|------------------|------------------|-----|------------------|------|------|-----|------|
| jméno databáze | jméno 1. položky | jméno 2. položky | ... | jméno X. položky | data | data | ... | data |

Pole obsahuje informace o jménech databáze a všech položek, kterých může být libovolný počet X. Dále obsahuje obsahy všech položek formátu 2, pro všechny záznamy v takovém pořadí, v jakém jsou v databázi uvedeny (tj. zleva doprava a shora dolů — viz příklad).

B. Pole TF(N)

| 0 | 1 | 2 | 3 | 4 | ... | X+2 | X+3 | ... | N |
|---------------|----------------------------|-------------------------|----------------|----------------|-----|----------------|------|-----|------|
| počet záznamů | počet položek v 1. záznamu | počet položek formátu 2 | formát 1. pol. | formát 2. pol. | ... | formát X. pol. | data | ... | data |

Pole obsahuje informace o struktuře databáze (velikost a formáty) a obsahu všech položek formátu 1.

Příklad uložení databáze v polích:

| | | | |
|----------------|--------|----------|-----------|
| Databáze: LIDÉ | JMÉNO | PŘÍJMENÍ | TELEFON |
| Jana Ivoš | Krátká | Pech | 3227 4138 |

Uložení:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-------|------|-------|----------|---------|------|--------|------|------|
| ADF\$ | LIDÉ | JMÉNO | PŘÍJMENÍ | TELEFON | Jana | Krátká | Ivoš | Pech |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|----|---|---|---|---|---|---|------|------|
| TF | 2 | 3 | 2 | 2 | 2 | 1 | 3227 | 4138 |

Dimenze obou polí má uživatel možnost upravovat na řádku 20 podle toho, jaký charakter budou mít ukládaná data. Musí mít ovšem stále na paměti, že:

pro ADF\$(n) ... n>X

pro TF(n) ... n>X+2

Jestliže tedy bude mít položky pouze formátu 1, potom může pole ADF\$ zkrátit na úkor pole TF a naopak. POZOR, při použití příkazu Z je třeba mít na zřeteli, jakou strukturu bude mít databáze zaváděná z magnetofonu.

STRUKTURA PROGRAMU

I když jazyk BASIC není optimální pro strukturované programování, při tvorbě programu bylo přihlíženo k tomu, aby program byl co nejčitelnější a nejsrozumitelnější. Proto je celý program dělen do programových celků, které začínají na řádcích XX000 a končí na řádcích XX990. Tyto programové celky mají charakter podprogramů, které vykonávají příslušné akce daného příkazu a jsou volány rozskokovou tabulkou na začátku programu.

Od čísla řádku 30000 jsou uloženy některé podprogramy společně pro činnost programu (definice obrazovky, ošetření chyb, definice české abecedy, servisní modul apod.).

Činnost programu a umístění příslušných podprogramů jsou zřejmé z grafických schémat.

OBSLUHA PROGRAMU RIDS

Program pracuje v interaktivním režimu a tudíž i nezaučený uživatel je schopen ho ovládat.

Obrazovka je rozdělena do dvou částí:

horní část — informační,
dolní část — konverzační.

V informační části jsou vypisovány výsledky prováděných příkazů a systémová hlášení. V konverzační části probíhá veškerá konverzace mezi systémem a uživatelem.

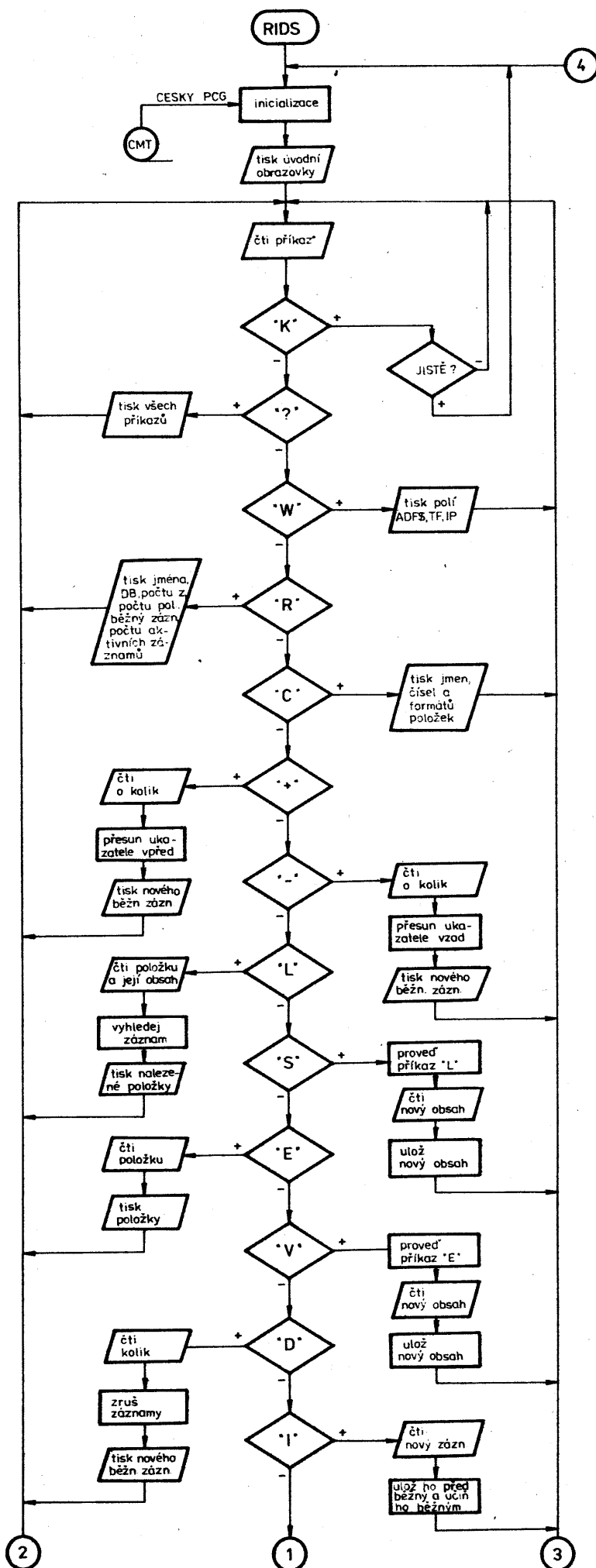
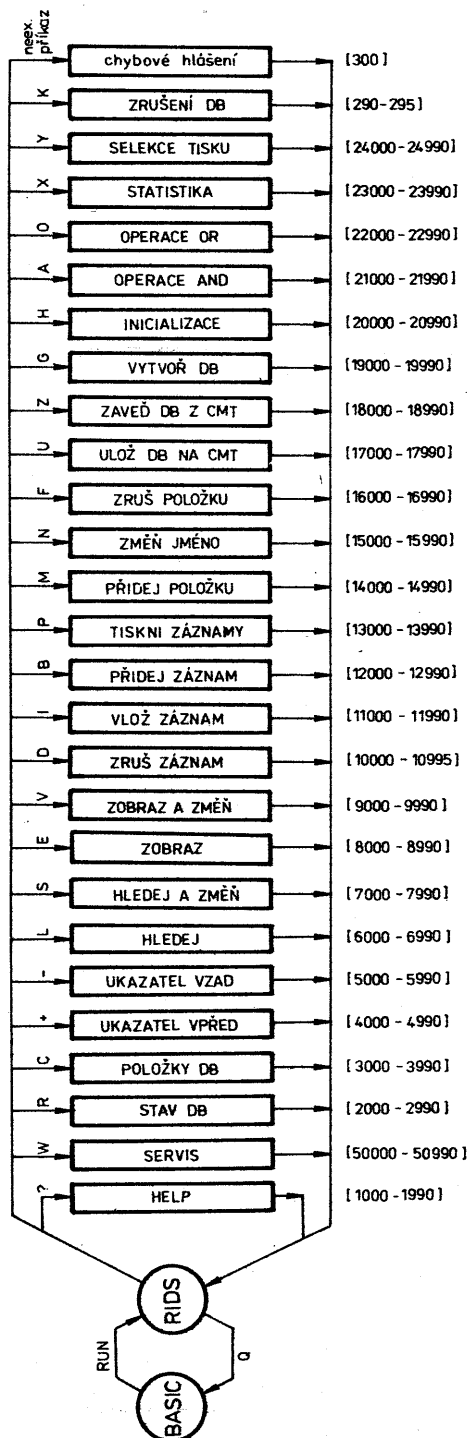
Po zavedení S-BASICu a zdrojového programu RIDS do paměti a po jeho spuštění se objeví na obrazovce úvodní hlášení, ve kterém je uživatel tážán, zda si přeje definovat českou abecedu. Jestliže ano, pak z CMT bude načten a proveden soubor ve strojovém kódu

„CESKY PCG“. Jestliže ne, pak tento soubor není potřebný k činnosti databáze. Po stisknutí příkazu ? je uživatel informován o všech jemu dostupných příkazech.

Poznámka: Prvním příkazem musí být vždy G nebo Z (viz dále), jinak systém reaguje chybovým hlášením. Po příkazu G nebo Z se doporučuje specifikovat formát tisku příkazem Y, protože implicitní nastavení tisku je omezeno pouze na tisk 1. položky bez uvedení jmen.

PŘÍKAZY

V konverzační části obrazovky systém indikuje příznakem „**“ stav, kdy očekává jednopísmenový kód příkazu.



Pro některé příkazy viz Implementační omezení.

A. Příkazy pro databázi

? — dostupné příkazy.

Vypíše všechny dostupné příkazy a jejich význam.

G — vytvoření struktury databáze.

Definuje se jméno databáze (tabulky),

počet položek záznamu (=počet sloupců tabulky), jména položek a formáty dat obsažených v položkách. Strukturu databáze lze později prostřednictvím jiných příkazů modifikovat.

K — zrušení databáze.

Zruší nejenom obsah databáze, ale i její strukturu.

Z — zavedení databáze.

Zavede dříve vytvořenou databázi

z CMT do paměti (obdoba Load). Je-li soubor s databází chráněn heslem, pak uživatel musí toto heslo specifikovat.

U — uložení databáze.

Stávající databáze se uloží na CMT (obdoba Save). Existuje možnost chránit ukládanou databázi heslem.

R — stav databáze.

Rozepíše stávající strukturu databáze (jméno, počet záznamů, počet položek, běžný záznam a počet aktivních záznamů).

Q — konec činnosti.

Ukončí činnost RIDS, používá se po předchozím použití U.

B. Příkazy pro záznamy

I — vložení záznamu.

Vsune nový záznam před běžný a nový záznam učiní běžným.

B — přidání záznamu.

Přidá nový záznam na konec databáze a učiní ho běžným.

Dn — zrušení záznamů.

Zruší n záznamů počínaje běžným (včetně) a běžným učiní následující nezrušený záznam. Je-li počet rušených záznamů větší než počet záznamů od běžného do konce databáze, pak se zruší pouze záznamy od běžného do konce databáze a běžným se stává záznam před prvním zrušeným. POZOR, ruší se pouze aktivní záznamy!

+n — přesun ukazatele běžného záznamu dopředu.

Učiní běžným ten záznam, který má pořadové číslo o n větší, než stávající běžný záznam. Dosáhne-li se konce databáze, stává se běžným poslední záznam.

-n — přesun ukazatele běžného záznamu dozadu.

Učiní běžným ten záznam, který má pořadové číslo o n menší, než stávající běžný záznam. Dosáhne-li se začátku databáze, stává se běžným první záznam.

Pn — výpis záznamů.

Funguje stejně jako příkaz D s tím rozdílem, že neruší záznamy, ale vypisuje je na obrazovku ve formátu zvoleném příkazem Y. Tisk se pozastavuje po každém stisknutí mezerníku, dalším stiskem mezerníku lze pokračovat. Zrušení tisku se provede klávesou Q.

H — inicializace množiny aktivních záznamů.

Převádí všechny záznamy do množiny aktivních záznamů.

A — operace AND.

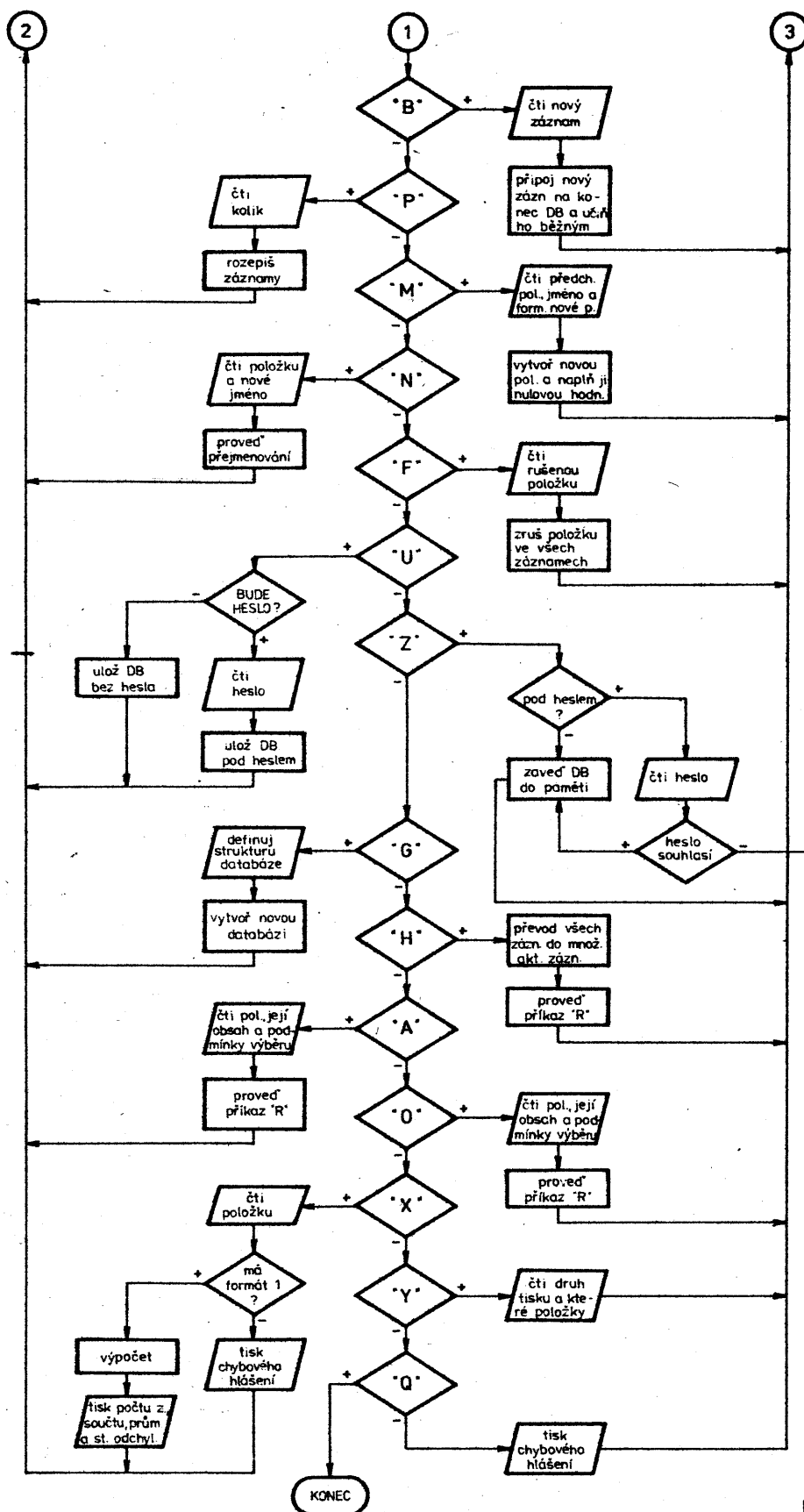
V množině aktivních záznamů ponechá pouze ty záznamy, jež splňují zadanou podmínku.

O — operace OR.

Do množiny aktivních záznamů přidá ty neaktivní, které splňují zadanou podmínku.

Y — selekce tisku.

Specifikuje formát tisku. POZOR, při tisku do řádku lze tisknout pouze 3 položky (vzhledem k omezené šířce obrazov-



ky) v libovolném pořadí, a to s maximální délkou řetězce 10 znaků (delší řetězec je zprava zkrácen — formát 2).

C. Příkazy pro položky

C — položky databáze.

Informace o položkách (čísla, jména a formáty).

M — přidání položky.

Přidá novou položku do struktury databáze. Obsah této položky je v každém záznamu nulový (viz Formát dat).

N — změna jména.

Změní jméno specifikované položky. Je-li číslo specifikované položky = 0, pak změní jméno databáze.

F — zrušení položky.

Zruší libovolnou položku databáze.

L — nalezení položky.

Nalezne první záznam s položkou, jejíž obsah definuje uživatel. Běžným záznamem se stává nalezený záznam anebo se nemění (nenajde-li se). Hledání začíná za běžným záznamem, pokračuje do konce databáze a dále pokračuje od začátku databáze do běžného záznamu.

S — vyhledání a změna.

Provádí totéž co příkaz L, s následnou změnou obsahu položky.

E — zobrazení položky.

Vypíše obsah libovolné položky běžného záznamu.

V — zobrazení a změna.

Provádí totéž co příkaz E, s následnou změnou obsahu položky.

X — statistika.

Provede výpočet statistických údajů na základě obsahu položky. POZOR, vyčíslovaná položka musí být formátu 1!

Vypočtené údaje jsou:

— počet záznamů (aktivních);

součet;

průměr;

standardní odchylka.

POZOR, vyčísluje pouze aktivní záznamy!

CHYBOVÁ HLÁŠENÍ

*** CHYBNÝ VSTUP ! ***

Indikuje chybu při operaci INPUT:

— formát 2 místo formátu 1; POZOR, ne naopak!

— chybné číslo položky;

— položka formátu 2 u příkazu X;

— nenumerická specifikace položky nebo formátu;

— formát položky = 1 nebo 2.

*** PRÁZDNÁ DATABÁZE ! ***

Pokus o provedení příkazu nad databází s nulovým počtem záznamů.

*** DATABÁZE NEVYTVOŘENA ! ***

Neproveden prvotní příkaz G nebo Z.

*** KAPACITA PŘEKROČENA ! ***

Další záznam do databáze není možný z důvodu překročení kapacity paměti. Použito u příkazů I, B, M.

*** ZRUŠTE DATABÁZI ! ***

U příkazu F, když uživatel ruší poslední položku databáze.

IMPLEMENTAČNÍ OMEZENÍ

a) Při zadávání parametrů pro příkazy se položka vždy specifikuje svým čís-

lem a ne jménem (lze zjistit příkazem C);

b) formát položky se vždy specifikuje svým číslem:

1... numerická data;

2... textová data.

c) nepoužívejte

SHIFT + **BREAK**,

došlo by ke zhroucení systému.

ČESKÁ ABECEDA

RIDS umožňuje používat velká i malá písmena a také háčky a čárky pomocí klávesy

GRAPH

Například, napsání slova UČENÍ se provede touto posloupností

U, **GRAPH**, **C**, **ALPHA**, **E**, **N**, **GRAPH**, **U**, **ALPHA**

kde **GRAPH** zapíná háčky a čárky a **ALPHA** vypíná háčky a čárky.

Protože písmena E a U mohou mít až tři varianty, je jejich přiřazení toto:

E = **GRAPH** + **E** **Ů** = **GRAPH** + **P**

Ě = **GRAPH** + **W** **Ú** = **GRAPH** + **U**

Výpis programu RIDS

```
10 REM *** RIDS ***
11 CLS:CURSOR 16,6:PRINT "R I D S".CURSOR
12 R 14,7:PRINT "JSS(C)-1987"
12 CURSOR 1,12:INPUT "DEFINOVAT CESKOU A
13 BECEDU ? (A/N) > ":OPX:OPX=LEFTX(OPX,1)
13 IF OPX="A" THEN PRINT:PRINT "JE NUTNY
14 SOUBOR NA CMT: >CESKY PCC< !":CURSOR 15
15,16:PRINT CHR$(X7F);" PLAY":LIMIT X7000:
16 LOAD "CESKY PCC":USR(X7000):LIMIT MAX:PO
17 KE X68CF,X3B,X1
14 ON ERROR GOTO 39000
15 COLOR ,7,0
20 ADF=500:NZF=500:NZD=200
25 DIM ADFX(ADF),TF(NZF),IP(NZD),TS(6)
30 CLS:OK=-1:AZ=0
35 GOSUB 33000
50 GOSUB 35000
53 MUSIC "A0.5"
55 PRINT " * ";
60 GET OPX:IF OPX="" THEN 60
62 PRINT OPX
65 IF OPX="W" THEN GOSUB 34000:GOSUB 500
66:GOTO 50
70 IF OPX="Q" THEN CONSOLE:CLS:END
80 IF OPX="?" THEN GOSUB 1000:GOTO 50
90 IF (OPX<>"G")*(OPX<>"Z")*(OK=-1) THEN
91 GOSUB 34000:CURSOR 5,5:PRINT " *** DATAB
92 AZE NEVYTVOŘENA ! ***":GOTO 50
110 IF OPX="R" THEN GOSUB 2000:GOTO 310
120 IF OPX="C" THEN GOSUB 3000:GOTO 310
130 IF OPX="+" THEN GOSUB 4000:GOTO 310
140 IF OPX="-" THEN GOSUB 5000:GOTO 310
150 IF OPX="L" THEN GOSUB 6000:GOTO 310
160 IF OPX="S" THEN GOSUB 7000:GOTO 310
170 IF OPX="E" THEN GOSUB 8000:GOTO 310
180 IF OPX="V" THEN GOSUB 9000:GOTO 310
190 IF OPX="D" THEN GOSUB 10000:GOTO 310
200 IF OPX="I" THEN GOSUB 11000:GOTO 310
210 IF OPX="B" THEN GOSUB 12000:GOTO 310
220 IF OPX="P" THEN GOSUB 13000:GOTO 310
230 IF OPX="M" THEN GOSUB 14000:GOTO 310
240 IF OPX="N" THEN GOSUB 15000:GOTO 310
250 IF OPX="F" THEN GOSUB 16000:GOTO 310
260 IF OPX="U" THEN GOSUB 17000:GOTO 310
270 IF OPX="Z" THEN GOSUB 18000:GOTO 310
280 IF OPX="G" THEN GOSUB 19000:GOTO 310
281 IF OPX="H" THEN GOSUB 20000:GOTO 310
282 IF OPX="A" THEN GOSUB 21000:GOTO 310
283 IF OPX="O" THEN GOSUB 22000:GOTO 310
284 IF OPX="X" THEN GOSUB 23000:GOTO 310
285 IF OPX="Y" THEN GOSUB 24000:GOTO 310
290 IF OPX="K" THEN PRINT:INPUT " POTVRD
291 TE! (A/N) > ":IX:IF LEFTX(IX,1)="A" THEN
30
```

```
295 IF (OPX="K")*(LEFTX(IX,1)<>"A") THEN
300 GOSUB 34000:CURSOR 5,6:PRINT CSX:GOT
310 OK=0:GOTO 50
1000 REM *** ?...HELP ***
1005 GOSUB 34000
1010 PRINT "PRIKAZ"
1020 PRINT "-----"
1040 PRINT " ?.....INFORMACE O PRIKAZEZ
H"
1050 PRINT " R.....STAV DATABAZE"
1060 PRINT " C.....POLOZKY DATABAZE"
1070 PRINT " +[N]...POSUV UKAZATELE VPRE
D"
1080 PRINT " -[N]...POSUV UKAZATELE VZAD
"
1090 PRINT " L.....NALEZENI ZAZNAMU"
1100 PRINT " S.....JAKO L SE ZMENOU"
1110 PRINT " E.....ZOBRAZENI OBSAHU POL
OZKY"
1120 PRINT " V.....JAKO E SE ZMENOU"
1130 PRINT " DENJ...ZRUSENI ZAZNAMU"
1140 PRINT " I.....VLOZENI ZAZNAMU PRED
BEZNY"
1150 PRINT " B.....PRIDANI ZAZNAMU NA K
ONEC"
1160 PRINT " PINJ...ROZEPSANI ZAZNAMU"
1162 CURSOR 2,20
1163 CONSOLE 18,6,2,36
1164 INPUT "DALSI PRIKAZ? (A/N) > ":OPX
1166 IF LEFTX(OPX,1)<>"A" THEN 1990
1167 GOSUB 34000
1168 PRINT "PRIKAZ"
1169 PRINT "-----"
1170 PRINT " M.....PRIDANI POLOZKY"
1180 PRINT " N.....PREJMENOVANI POLOZKY
"
1190 PRINT " F.....ZRUSENI POLOZKY"
1200 PRINT " U.....ULOZENI DATABAZE NA
CMT"
1210 PRINT " Z.....ZAVEDENI DATABAZE Z
CMT"
1220 PRINT " G.....VYTVORENI DATABAZE"
1230 PRINT " K.....ZRUSENI DATABAZE"
1240 PRINT " H.....AKTIVIZACE ZAZNAMU"
1250 PRINT " A.....AKTIVIZACE - OPERACI
AND"
1260 PRINT " O.....AKTIVIZACE - OPERACI
OR"
1270 PRINT " X.....STATISTIKA DAT"
1280 PRINT " Y.....SELEKCE TISKU"
1290 PRINT " Q.....KONEC PRACE S DATABA
ZI"
1990 RETURN
2000 REM *** R...STAV DATABAZE ***
2005 GOSUB 34000
2010 PRINT "DATABAZE : ";ADFX(0)
2020 PRINT "ZAZNAMU : ";TF(0)
2030 PRINT "POLOZEK : ";TF(1)
2040 PRINT
2050 PRINT "BEZNY ZAZNAM : ";AZ;" "
2060 PRINT "-----"
2070 POC=0
2075 IF TF(0)=0 THEN 2110
2080 FOR I=1 TO TF(0)
2090 IF IP(I)=0 THEN POC=POC+1
2100 NEXT I
2110 PRINT "AKTIVNICH ZAZNAMU : ";POC
2990 RETURN
3000 REM *** C...POLOZKY ***
3005 GOSUB 34000
3010 FOR I=1 TO TF(1)
3020 PRINT I:" ";ADFX(I);TAB(26);"FORMA
T =";ABS(TF(I+2))
3030 NEXT I
3990 RETURN
4000 REM *** +...POSUV VPRED ***
4005 IF TF(0)=0 THEN GOTO 36000
4010 INPUT "O KOLIK >":KOLIK
4015 IF (KOLIK<0)+(KOLIK>INT(KOLIK)) TH
EN GOSUB 37000:GOTO 4010
4020 AZ=AZ+KOLIK
4030 IF AZ>TF(0) THEN AZ=TF(0)
4990 KOLIK=1:GOTO 13020
5000 REM *** -...POSUV VZAD ***
5005 IF TF(0)=0 THEN GOTO 36000
5010 INPUT "O KOLIK >":KOLIK
5015 IF (KOLIK<0)+(KOLIK>INT(KOLIK)) TH
EN GOSUB 37000:GOTO 5010
5020 AZ=AZ-KOLIK
5030 IF AZ<1 THEN AZ=1
5990 GOTO 4990
6000 REM *** L...VYHLEDANI ZAZNAMU ***
6005 IF TF(0)=0 THEN GOTO 36000
6010 INPUT "POLOZKA >":SL
```

```

6020 IF (SL>TF(1))+(SL<1)+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 6010
6030 IF ABS(TF(SL+2))=1 THEN INPUT "OBŠA
H >";PR:GOTO 6040
6035 INPUT "OBŠAH >";PRX
6040 PP=0:KOLIK=0
6045 IF SL=1 THEN 6075
6050 FOR I=1 TO SL-1
6060 IF ABS(TF(I+2))=2 THEN PP=PP+1
6070 NEXT I
6075 POC=PP
6080 IF ABS(TF(SL+2))=1 THEN 6200
6090 PP=PP+TF(1)+1+AZ*TF(2)
6100 IF TF(0)<AZ+1 THEN 6140
6105 FOR I=AZ+1 TO TF(0)
6110 IF ADFX(PP)=PRX THEN AZ=I:GOTO 4990
6120 PP=PP+TF(2)
6130 NEXT I
6140 PP=POC+TF(1)+1
6145 IF AZ<1 THEN 6190
6150 FOR I=1 TO AZ
6160 IF ADFX(PP)=PRX THEN AZ=I:GOTO 4990
6170 PP=PP+TF(2)
6180 NEXT I
6190 GOTO 6990
6200 PP=2+TF(1)+AZ*(TF(1)-TF(2))+SL-POC
6210 IF TF(0)<AZ+1 THEN 6250
6215 FOR I=AZ+1 TO TF(0)
6220 IF TF(PP)=PR THEN AZ=I:GOTO 4990
6230 PP=PP+TF(1)-TF(2)
6240 NEXT I
6250 PP=2+TF(1)+SL-POC
6255 IF AZ<1 THEN 6990
6260 FOR I=1 TO AZ
6270 IF TF(PP)=PR THEN AZ=I:GOTO 4990
6280 PP=PP+TF(1)-TF(2)
6290 NEXT I
6990 RETURN
7000 REM *** S...VYHLEDANI A ZMENA ***
7005 IF TF(0)=0 THEN GOTO 36000
7010 GOSUB 6000
7015 IF KOLIK=0 THEN 7990
7017 CURSOR 2,22
7019 CONSOLE 18,6,2,36
7020 IF ABS(TF(SL+2))=1 THEN INPUT "NOVY
OBŠAH >";TF((AZ-1)*(TF(1)-TF(2))+SL+TF(
1)+2-POC):GOTO 7990
7030 INPUT "NOVY OBŠAH >";ADFX((AZ-1)*TF
(2)+POC+1+TF(1))
7990 RETURN
8000 REM *** E...ZOBRAZENÍ POLOŽKY ***
8005 IF TF(0)=0 THEN GOTO 36000
8010 INPUT "ČÍSLO POLOŽKY >";SL
8015 IF (SL>TF(1))+(SL<1)+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 8010
8020 POC=0
8025 IF SL=1 THEN 8055
8030 FOR I=1 TO SL-1
8040 IF ABS(TF(I+2))=2 THEN POC=POC+1
8050 NEXT I
8055 GOSUB 34000
8060 IF ABS(TF(SL+2))=1 THEN PRINT "OBŠA
H =";TF((AZ-1)*(TF(1)-TF(2))+SL+TF(1)+2-
POC):GOTO 8990
8070 PRINT "OBŠAH = ";ADFX((AZ-1)*TF(2)+
POC+TF(1)+1)
8990 RETURN
9000 REM *** V...ZOBRAZENÍ A ZMENA ***
9005 IF TF(0)=0 THEN GOTO 36000
9010 GOSUB 8000
9990 GOTO 7017
10000 REM *** D...RUŠENÍ ZAZNAMU ***
10005 IF TF(0)=0 THEN 36000
10010 INPUT "KOLIK >";KOLIK
10015 IF (KOLIK<0)+(KOLIK<>INT(KOLIK)) T
HEN GOSUB 37000:GOTO 10010
10018 IF KOLIK=0 THEN 10990
10020 IF KOLIK=TF(0)-AZ+1 THEN KOLIK=TF
(0)-AZ+1:GOTO 10150
10030 POC=(AZ-1)*(TF(1)-TF(2))+TF(1)+3
10040 PP=KOLIK*(TF(1)-TF(2))
10045 IF PP<1 THEN 10090
10050 FOR I=1 TO (TF(0)-AZ)*(TF(1)-TF(2)
)
10060 TF(POC)=TF(POC+PP)
10070 POC=POC+1
10080 NEXT I
10090 POC=(AZ-1)*TF(2)+TF(1)+1
10100 PP=KOLIK+TF(2)
10105 IF PP<1 THEN 10141
10110 FOR I=1 TO (TF(0)-AZ)*TF(2)
10120 ADFX(POC)=ADFX(POC+PP)
10130 POC=POC+1
10140 NEXT I
10141 IF KOLIK=1 THEN 10150
10142 FOR I=1 TO KOLIK-1
10144 IP(AZ+I)=IP(AZ+KOLIK+I)
10146 NEXT I
10150 IF AZ+KOLIK>TF(0) THEN AZ=AZ-1
10152 TF(0)=TF(0)-KOLIK
10990 GOTO 2000
11000 REM *** I...VLOZ ZAZNAM ***
11005 IF (((TF(0)+1)*(TF(1)-TF(2))+TF(1)
+2)>NTF)+((TF(0)+1)*TF(2)+TF(1))>ADF) T
HEN GOSUB 38000:GOTO 11990
11008 IF AZ=0 THEN AZ=1
11010 POC=TF(0)*(TF(1)-TF(2))+TF(1)+2
11020 PP=TF(1)-TF(2)
11025 IF (TF(0)-AZ+1)*(TF(1)-TF(2))<1 TH
EN 11070
11030 FOR I=1 TO (TF(0)-AZ+1)*(TF(1)-TF(
2))
11040 TF(POC+PP)=TF(POC)
11050 POC=POC+1
11060 NEXT I
11070 POC=TF(0)*TF(2)+TF(1)
11080 PP=TF(2)
11085 IF TF(2)*(TF(0)-AZ+1)<1 THEN 11130
11090 FOR I=1 TO TF(2)*(TF(0)-AZ+1)
11100 ADFX(POC+PP)=ADFX(POC)
11110 POC=POC+1
11120 NEXT I
11130 POC=POC+1
11140 PP=(AZ-1)*(TF(1)-TF(2))+TF(1)+3
11145 GOSUB 34000
11150 FOR I=1 TO TF(1)
11160 PRINT I;";
11165 IF LEFTX(JSX,1)="" THEN PRINT ADF
X(I);
11170 IF ABS(TF(I+2))=1 THEN INPUT ">";
TF(PP):PP=PP+1:GOTO 11200
11180 INPUT ">";ADFX(POC):POC=POC+1
11200 NEXT I
11201 IF TF(0)-AZ<0 THEN 11210
11204 FOR I=0 TO TF(0)-AZ
11206 IP(TF(0)-I+1)=IP(TF(0)-I)
11208 NEXT I
11210 IP(AZ)=0
11220 TF(0)=TF(0)+1
11990 RETURN
12000 REM *** B...PRIDEJ ZAZNAM ***
12005 IF (((TF(0)+1)*(TF(1)-TF(2))+TF(1)
+2)>NTF)+((TF(0)+1)*TF(2)+TF(1))>ADF) THE
N GOSUB 38000:GOTO 11990
12010 POC=TF(1)+TF(2)*TF(0)+1
12020 PP=TF(1)+TF(0)*(TF(1)-TF(2))+3
12030 AZ=TF(0)+1
12990 GOTO 11145
13000 REM *** P...TISK ZAZNAMU ***
13005 IF TF(0)=0 THEN GOTO 36000
13010 INPUT "KOLIK >";KOLIK
13015 IF (KOLIK<0)+(KOLIK<>INT(KOLIK)) T
HEN GOSUB 37000:GOTO 13010
13020 P2=(AZ-1)*TF(2)+TF(1)+1
13040 PP=(AZ-1)*(TF(1)-TF(2))+TF(1)+3:P1
=0
13045 GOSUB 34000
13047 IF (LEFTX(OLX,1)="" AND (LEFTX(JSX,1)
)="" AND (LEFTX(OLX,1)="" AND (LEFTX(JSX,1)
)="" THEN PRINT SPC(3);:FOR I=1 TO TS(
0):PRINT LEFTX(ADFX(TS(I)),9);TAB(I*10+5
);:NEXT I:PRINT "
";CONSOLE 3,16,2,36
13050 FOR I=AZ TO TF(0)
13055 IF IP(I)=1 THEN P2=P2+TF(2):PP=PP+
TF(1)-TF(2):GOTO 13120
13057 IF P1=KOLIK THEN 13130
13058 IF LEFTX(OLX,1)="" THEN 13102
13060 FOR J=1 TO TF(1)
13065 IF TF(J+2)=1 THEN PP=PP+1:GOTO 131
00
13067 IF TF(J+2)=2 THEN P2=P2+1:GOTO 131
00
13070 PRINT J;";
13072 IF LEFTX(JSX,1)="" THEN PRINT ADF
X(J);";
13074 PRINT ">";
13076 IF ABS(TF(J+2))=1 THEN PRINT TF(PP
):PP=PP+1:GOTO 13100
13078 PRINT " ";ADFX(P2):P2=P2+1
13100 NEXT J
13101 GOTO 13114
13102 PRINT SPC(3);:FOR J=1 TO TS(0)
13104 IF TF(TS(J)+2)=1 THEN PRINT TF(PP
-TS(J+3)+TS(J)-1):GOTO 13107
13106 PRINT LEFTX(ADFX(P2+TS(J+3)),9);
13107 PRINT TAB(J*10+5);
13108 NEXT J
13110 PRINT
13112 P2=P2+TF(2):PP=PP+TF(1)-TF(2)
13114 AZ=AZ+1
13116 IF LEFTX(OLX,1)<>"A" THEN PRINT "-
"
13117 P1=P1+1
13118 GET IX:IF (LEFTX(IX,1)="" AND (LEFTX
(OPX,1)="" AND (LEFTX(IX,1)="" AND (LEFTX
13119 IF LEFTX(IX,1)="" THEN FOR I=1 TO
500:NEXT I:GOTO 13130
13120 NEXT I
13130 IF P1<>0 THEN AZ=AZ-1
13990 RETURN
14000 REM *** A...PRIDANI POLOŽKY ***
14010 INPUT "PŘEDCHÁZEJÍCÍ POLOŽKA >";SL
14015 IF (SL<0)+(SL<>INT(SL))+(SL>TF(1))
THEN GOSUB 37000:GOTO 14010
14020 PP=0
14025 IF SL<1 THEN 14060
14030 FOR I=1 TO SL
14040 IF ABS(TF(I+2))=2 THEN PP=PP+1
14050 NEXT I
14060 INPUT "JENOV NOVE POLOŽKY >";PRX
14070 INPUT "FORMAT NOVE POLOŽKY >";PR
14075 IF (PR<>1)*(PR<>2) THEN GOSUB 3700
0:GOTO 14070
14076 IF PR=1 THEN IF (TF(0)*(TF(1)-TF(2)
))+TF(1)+2+TF(0))>NTF THEN GOSUB 38000:
GOTO 14990
14077 IF (TF(0)*TF(2)+TF(1)+TF(0))>ADF T
HEN GOSUB 38000:GOTO 14990
14080 IF PR=1 THEN POC=TF(1)+2+TF(1)-TF
(2))*TF(0):PO=TF(0):GOTO 14195
14090 POC=TF(1)+TF(0)*TF(2):PO=TF(0)
14095 IF TF(0)=0 THEN 14190
14100 FOR I=1 TO TF(0)
14105 IF TF(2)-PP<1 THEN 14140
14110 FOR J=1 TO TF(2)-PP
14120 ADFX(POC+PQ)=ADFX(POC):POC=POC+1
14130 NEXT J
14140 ADFX(POC+PQ)="" :PQ=PQ-1
14145 IF PP<1 THEN 14180
14150 FOR J=1 TO PP
14160 ADFX(POC+PQ)=ADFX(POC):POC=POC+1
14170 NEXT J
14180 NEXT I
14190 TF(2)=TF(2)+1:GOTO 14290
14195 IF TF(0)=0 THEN 14290
14200 FOR I=1 TO TF(0)
14205 IF TF(1)-TF(2)-SL+PP<1 THEN 14240
14210 FOR J=1 TO TF(1)-TF(2)-SL+PP
14220 TF(POC+PQ)=TF(POC):POC=POC+1
14230 NEXT J
14240 TF(POC+PQ)=0:PQ=PQ-1
14245 IF SL-PP<1 THEN 14280
14250 FOR J=1 TO SL-PP
14260 TF(POC+PQ)=TF(POC):POC=POC+1
14270 NEXT J
14280 NEXT I
14290 TF(1)=TF(1)+1
14300 POC=(TF(1)-TF(2))*TF(0)+TF(1)+1
14305 IF POC-SL-2<1 THEN 14350
14310 FOR I=1 TO POC-SL-2
14320 TF(POC+1)=TF(POC)
14330 POC=POC+1
14340 NEXT I
14350 POC=TF(2)*TF(0)+TF(1)-1
14355 IF POC<SL+1 THEN 14400
14360 FOR I=1 TO POC-SL
14370 ADFX(POC+1)=ADFX(POC)
14380 POC=POC+1
14390 NEXT I
14400 TF(SL+3)=-PR
14410 ADFX(SL+1)=PRX
14990 GOTO 2000
15000 REM *** N...ZMENA JMENA POLOŽKY ***
15010 INPUT "PŘEJMENOVÁVÁNÍ POLOŽKA >";S
L
15015 IF (SL<0)+(SL>TF(1))+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 15010
15020 INPUT "NOVE JMENO >";PRX
15030 ADFX(SL)=PRX
15990 RETURN
16000 REM *** F...RUŠENÍ POLOŽKY ***
16006 IF TF(1)-1<1 THEN GOSUB 34000:CURS
OR 8,6:PRINT "*** ZRUŠTE DATABAZI ! ***"
:RETURN
16010 INPUT "RUŠENÍ POLOŽKA >";SL
16015 IF (SL<1)+(SL>TF(1))+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 16010
16020 PP=0
16025 IF SL=1 THEN 16080
16030 FOR I=1 TO SL-1
16040 IF ABS(TF(I+2))=2 THEN PP=PP+1
16050 NEXT I
16080 IF ABS(TF(SL+2))=1 THEN POC=TF(1)+
2-PP+SL:PQ=1:GOTO 16195
16090 POC=TF(1)+PP+1:PQ=1

```

```

16095 IF TF(0)=0 THEN 16190
16100 FOR I=1 TO TF(0)
16105 IF TF(2)-1<1 THEN 16140
16110 FOR J=1 TO TF(2)-1
16120 ADFX(POC)=ADFX(POC+PQ):POC=POC+1
16130 NEXT J
16140 PQ=PQ+1
16180 NEXT I
16190 TF(2)=TF(2)-1:GOTO 16290
16195 IF TF(0)=0 THEN 16290
16200 FOR I=1 TO TF(0)
16205 IF TF(1)-TF(2)-1<1 THEN 16280
16210 FOR J=1 TO TF(1)-TF(2)-1
16220 TF(POC)=TF(POC+PQ):POC=POC-1
16230 NEXT J
16240 PQ=PQ-1
16280 NEXT I
16290 TF(1)=TF(1)-1
16300 POC=SL+2
16305 IF (TF(1)-TF(2))*TF(0)+TF(1)-SL+1<1 THEN 16350
16310 FOR I=1 TO (TF(1)-TF(2))*TF(0)+TF(1)-SL+1
16320 TF(POC)=TF(POC+1)
16330 POC=POC+1
16340 NEXT I
16350 POC=SL
16355 IF TF(2)*TF(0)+TF(1)-SL+1<1 THEN 16990
16360 FOR I=1 TO TF(2)*TF(0)+TF(1)-SL+1
16370 ADFX(POC)=ADFX(POC+1)
16380 POC=POC+1
16390 NEXT I
16990 GOTO 2000
17000 REM *** U...SAVE ***
17005 IF TF(0)=0 THEN GOTO 36000
17006 INPUT "CHRANIT ZAPIS? (A/N) >";OPX
17007 IF LEFTX(OPX,1)="N" THEN PRX="";GOTO 17010
17008 IF LEFTX(OPX,1)<>"A" THEN PRINT:PR
INT CSX:PRINT:GOTO 17006
17009 INPUT "HESLO >";PRX:CLS
17010 GOSUB 34000
17020 CURSOR 13,5:PRINT CHR(X7F);" RECO
RD.PLAY"
17022 J=TF(0)*(TF(1)-TF(2))+2+TF(1)
17024 K=TF(0)*TF(2)+TF(1)
17025 WOPEN ADFX(0)
17030 PRINT/T PRX
17040 PRINT/T J
17050 PRINT/T K
17060 FOR I=0 TO J
17100 PRINT/T TF(I)
17110 NEXT I
17140 FOR I=0 TO K
17150 PRINT/T ADFX(I)
17170 NEXT I
17240 CLOSE
17250 CLS
17990 RETURN
18000 REM *** Z...LOAD ***
18002 IF OK=0 THEN INPUT "EXISTUJICI DAT
ABAZI ZRUSIT?(A/N) >";OPX:IF LEFTX(OPX,1)
<>"A" THEN 18990
18005 INPUT "JMENO DATABASE Z CMT >";OPX
18008 GOSUB 34000
18009 CURSOR 16,5:PRINT CHR(X7F);" PLAY
"
18010 OPEN OPX
18011 INPUT/T PRX
18012 IF PRX<>""; THEN PRINT:INPUT "
HESLO? >";OPX:CLS:IF PRX<>OP
X THEN CLOSE:GOTO 10
18013 INPUT/T J
18016 INPUT/T K
18020 FOR I=0 TO J
18030 INPUT/T TF(I)
18040 NEXT I
18050 FOR I=0 TO K
18060 INPUT/T ADFX(I)
18070 NEXT I
18080 CLOSE
18090 FOR I=0 TO TF(0):IP(I)=0:NEXT I
18095 FOR I=3 TO TF(1)+2:TF(I)=ABS(TF(I)
):NEXT I
18100 AZ=1:OK=0:TF(3)=-TF(3)
18110 JSX="A":OLX="N"
18990 GOTO 2000
19000 REM *** G...VYTVORENI DATABASE ***
19002 IF OK=-1 THEN 19008
19004 INPUT "EXISTUJICI DATABASE ZRUSIT?
(A/N) >";OPX

```

```

19006 IF LEFTX(OPX,1)<>"A" THEN 19990
19008 PRINT:PRINT "CEKEJTE PROSIM !"
19010 FOR I=0 TO ADF
19020 ADFX(I)=""
19030 NEXT I
19040 FOR I=0 TO NTF
19050 TF(I)=0
19060 NEXT I
19061 FOR I=0 TO NZD
19062 IP(I)=0
19063 NEXT I
19065 CONSOLE 1,23,1,38:CLS:CURSOR 2,1:C
ONSOLE 1,23,2,36
19070 INPUT "JMENO NOVE DATABASE....."
:PRX
19080 INPUT "POTVRDTE VYTVORENI (A/N)...
":OPX
19090 IF LEFTX(OPX,1)<>"A" THEN OK=-1:GO
TO 30
19100 INPUT "POCET POLOZEK DATABASE...."
:TF(1)
19101 IF (TF(1)<1)+(TF(1)<>INT(TF(1))) T
HEN GOSUB 37000:GOTO 19100
19102 INPUT "POCET POLOZEK FORMATU 2..."
:TF(2)
19103 IF (TF(2)<0)+(TF(2)<>INT(TF(2))) T
HEN GOSUB 37000:GOTO 19102
19104 IF (ADF-TF(1)-TF(2)<0)+(NTF-2-2*TF
(1)+TF(2)<0) THEN PRINT:PRINT SPC(2);KSX
:PRINT:GOTO 19100
19110 ADFX(0)=PRX:POC=0
19120 FOR I=1 TO TF(1)
19130 PRINT:PRINT "POLOZKA";I
19140 INPUT " JMENO...";ADFX(I)
19150 INPUT " FORMAT...";TF(I+2)
19155 IF (TF(I+2)<1)+(TF(I+2)<>2) THEN
GOSUB 37000:GOTO 19150
19157 IF TF(I+2)=2 THEN POC=POC+1
19160 NEXT I
19170 PRINT:INPUT "NEJAKE OPRAVY? (A/N)...
":OPX
19180 IF LEFTX(OPX,1)<>"A" THEN 19270
19190 INPUT "OPRAVOVANA POLOZKA....."
:SL
19195 IF (SL<>INT(SL))+(SL<1)+(SL>TF(1))
THEN GOSUB 37000:GOTO 19190
19200 INPUT "ZMENIT JMENO? (A/N)....."
:OPX
19210 IF LEFTX(OPX,1)<>"A" THEN 19230
19220 INPUT " NOVE JMENO....."
:ADFX(SL)
19230 INPUT "ZMENIT FORMAT? (A/N)....."
:OPX
19240 IF LEFTX(OPX,1)<>"A" THEN 19260
19250 INPUT " NOVY FORMAT....."
:PR
19255 IF (PR<1)*(PR<2) THEN GOSUB 3700
0:GOTO 19250
19256 IF (TF(SL+2)=2)*(PR=1) THEN POC=PO
C-1:GOTO 19259
19257 IF (TF(SL+2)=1)*(PR=2) THEN POC=PO
C+1
19258 IF (ADF-TF(1)-POC<0)+(NTF-2-2*TF(1)
)+POC<0) THEN PRINT:PRINT SPC(2);KSX:PRI
NT:GOTO 19100
19259 TF(SL+2)=PR
19260 GOTO 19170
19270 CLS:CONSOLE:CURSOR 1,17:PRINT [1,]
"===== NOVA DATABASE =====
"
19990 TF(3)=-TF(3):TF(2)=POC:AZ=0:RETURN
20000 REM *** H...AKTIVIZACE ZAZNAMU ***
20005 IF TF(0)=0 THEN GOTO 36000
20010 FOR I=1 TO TF(0)
20020 IP(I)=0
20030 NEXT I
20990 GOTO 2000
21000 REM *** A...OPERACE AND ***
21005 IF TF(0)=0 THEN GOTO 36000
21010 GOSUB 30000
21030 FOR I=1 TO TF(0)
21040 IF IP(I)=0 THEN GOSUB 31000
21050 POC=POC+TF(2):PP=PP+TF(1)-TF(2)
21060 NEXT I
21990 GOTO 2000
22000 REM *** O...OPERACE OR ***
22005 IF TF(0)=0 THEN GOTO 36000
22010 GOSUB 30000
22040 FOR I=1 TO TF(0)
22050 IF IP(I)=1 THEN GOSUB 31000
22060 POC=POC+TF(2):PP=PP+TF(1)-TF(2)
22070 NEXT I
22990 GOTO 2000
23000 REM *** X...STATISTIKA ***
23005 IF TF(0)=0 THEN GOTO 36000
23010 INPUT "POLOZKA >";SL
23015 IF (SL<1)+(SL>TF(1))+(SL<>INT(SL))

```

```

+(ABS(TF(SL+2))=2) THEN GOSUB 37000:GOTO
23010
23030 INPUT "TAKE STANDARTNI ODCHYLKU? (
A/N) >";OPX
23035 GOSUB 34000
23040 PRINT ADFX(SL):PRINT
23042 PP=0
23043 IF SL-1<1 THEN 23050
23044 FOR I=1 TO SL-1
23046 IF ABS(TF(I+2))=1 THEN PP=PP+1
23048 NEXT I
23050 P1=0:P2=0:P3=0:POC=TF(1)+3+PP
23060 FOR I=1 TO TF(0)
23070 IF IP(I)=1 THEN 23100
23080 P1=P1+1
23090 P2=P2+TF(POC)
23100 POC=POC+TF(1)-TF(2)
23110 NEXT I
23120 PRINT " ZAZNAMU =";P1
23125 PRINT " SOUCET =";P2
23130 IF P1=0 THEN 23990
23140 P2=P2/P1
23150 PRINT " PRUMER =";P2
23160 IF LEFTX(OPX,1)<>"A" THEN 23990
23165 POC=TF(1)+3+PP
23170 FOR I=1 TO TF(0)
23180 IF IP(I)=1 THEN 23200
23190 P3=P3+(TF(POC)-P2)^2
23200 POC=POC+TF(1)-TF(2)
23210 NEXT I
23220 PRINT:PRINT " STANDARTNI ODCHYLK
A =";SQR(P3/P1)
23990 RETURN
24000 REM *** Y...SELEKCE TISKU ***
24005 CONSOLE 1,23,1,38:CLS:CURSOR 2,1:C
ONSOLE 1,23,2,36
24010 INPUT "KOLIK ZOBRAZIT POLOZEK.....
":SL
24011 IF (SL<0)+(SL>TF(1))+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 24010
24012 IF SL=0 THEN OLX="N":TF(3)=ABS(TF(
3)):GOTO 24090
24013 INPUT "DO RADKU? (A/N)....."
:OLX:PRINT
24014 IF (LEFTX(OLX,1)="A")*(SL>3) THEN
SL=3
24020 FOR I=1 TO TF(1)
24030 TF(I+2)=ABS(TF(I+2))
24040 NEXT I
24042 FOR I=0 TO 6:TS(I)=0:NEXT I
24045 P1=0
24050 FOR I=1 TO SL
24060 PRINT "CISLO";I:" POLOZKY";
24065 INPUT ".....":PR
24066 IF (PR<1)+(LEFTX(OLX,1)<>"A")*(PR<
=P1)+(PR>TF(1))+(PR<>INT(PR)) THEN GOSUB
37000:GOTO 24060
24067 P1=PR
24068 IF LEFTX(OLX,1)<>"A" THEN 24074
24069 IF PR-1<1 THEN 24073
24070 FOR J=1 TO PR-1
24071 IF ABS(TF(J+2))=2 THEN TS(I+3)=TS(
I+3)+1
24072 NEXT J
24073 TS(I)=PR:TS(0)=TS(0)+1
24074 TF(PR+2)=ABS(TF(PR+2))
24075 IF (LEFTX(OLX,1)<>"A")*(P1=TF(1))
THEN 24090
24080 NEXT I
24090 PRINT:INPUT "ZOBRAZIT JMENA POLOZE
K? (A/N)...";JSX
24100 CLS:CONSOLE:CURSOR 1,17:PRINT [1,]
"=====
"
24990 RETURN
30000 REM *** SUBROUTINA AKTIVACE 1 ***
30010 INPUT "POLOZKA >";SL
30015 IF (SL<1)+(SL>TF(1))+(SL<>INT(SL))
THEN GOSUB 37000:GOTO 30010
30020 INPUT "<> NEBO = (KTERA Z NICH?)
>";OPX
30025 IF (LEFTX(OPX,1)<>"")*(LEFTX(OPX,
1)<>"")*(LEFTX(OPX,1)<>"") THEN GOSUB
37000:GOTO 30020
30030 IF ABS(TF(SL+2))=1 THEN INPUT "HOD
NOTA >";PR:GOTO 30040
30035 INPUT "HODNOTA >";PRX
30040 POC=0:PP=0
30050 FOR I=1 TO SL
30060 IF ABS(TF(I+2))=2 THEN POC=POC+1:G
OTO 30080
30070 PP=PP+1
30080 NEXT I
30090 PP=TF(1)+2+PP
30100 POC=TF(1)+POC
30990 RETURN

```



```

31000 REM *** SUBROUTINA AKTIVACE 2 ***
31010 IF ABS(TF(SL+2))=2 THEN 31070
31020 IF LEFTX(OPX,1)="" THEN IF PR>TF(
PP) THEN 31110
31030 IF LEFTX(OPX,1)="" THEN IF PR=TF(
PP) THEN 31110
31040 IF LEFTX(OPX,1)="" THEN IF PR<TF(
PP) THEN 31110
31050 GOTO 31990
31070 IF LEFTX(OPX,1)="" THEN IF PR>LE
FTX(ADF(POC),LEN(PRX)) THEN 31110
31080 IF LEFTX(OPX,1)="" THEN IF PR=LE
FTX(ADF(POC),LEN(PRX)) THEN 31110
31090 IF LEFTX(OPX,1)="" THEN IF PR<LE
FTX(ADF(POC),LEN(PRX)) THEN 31110
31100 GOTO 31990
31110 IP(I)=0:RETURN
31990 IP(I)=1:RETURN
32000 REM *** WAITING ***
32010 GET IX:IF (LEFTX(IX,1)<"")*(LEFT
X(IX,1)<"Q") THEN 32010
32990 RETURN
33000 REM *** UVOD ***
33005 CONSOLE:CURSOR 0,0
33010 CSX=""
33020 KSX=""
33030 PRINT [1,1] CSX
33040 FOR I=1 TO 16:PRINT [1,1] KSX:NEXT
I
33050 PRINT [1,1] CSX
33060 FOR I=1 TO 6:PRINT [1,1] KSX:NEXT
I
33070 PRINT [1,1] CSX
33080 CSX=""
33090 KSX="I"
33100 CURSOR 12,3:PRINT CSX
33110 FOR I=4 TO 11:CURSOR 12,I:PRINT KS
X:NEXT I
33120 CURSOR 12,12:PRINT CSX
33130 CURSOR 12,9:PRINT CSX
33140 CURSOR 16,5:PRINT "DATABASE";
33150 CURSOR 16,7:PRINT "R I D S";
33160 CURSOR 14,10:PRINT "V3.2 (4.8.87)"
33170 CURSOR 15,11:PRINT "(C) SVEHLA";
33180 CURSOR 15,14:PRINT "STISKNI ?";
33190 KSX="*** KAPACITA PREKROCENA ! ***"
33200 PSX=""
33210 CSX=""
33220 CHYBY VSTUP ! ***
33990 RETURN
34000 REM *** KONSOLA HORNÍ ***
34010 CURSOR 2,1
34020 CONSOLE 1,16,2,36
34030 CLS
34990 RETURN
35000 REM *** KONSOLA DOLNÍ ***
35010 CURSOR 2,18
35020 CONSOLE 18,6,2,36
35030 CLS
35990 RETURN
36000 REM *** PRAZDNA DATABASE ***
36005 GOSUB 34000
36010 CURSOR 5,5:PRINT PSX
36990 RETURN
37000 REM *** CHYBY VSTUP ***
37010 PRINT:PRINT SPC(5);CSX:PRINT
37990 RETURN
38000 REM *** KAPACITA PREKROCENA ***
38005 GOSUB 34000
38010 CURSOR 5,3:PRINT KSX
38990 RETURN
39000 REM *** OŠETŘENÍ SYST. CHYBY ***
39010 GOSUB 37000
39990 RESUME
50000 CLS:FOR I=0 TO NTF
50001 PRINT ADF(I);TAB(25);IP(I);TF(I)
50005 GET IX:IF LEFTX(IX,1)="" THEN GOS
UB 32000
50007 IF LEFTX(IX,1)="" THEN 50990
50010 NEXT I
50990 RETURN

```

Definice českého generátoru znaků

```

F000=0B E0 11 80 C1 21 78 F0 /m0. a!xP
F008=01 30 00 ED B0 11 C0 C1 /...M .a
F010=01 10 00 ED B0 11 E8 C1 /...M .Xa
F018=01 08 00 ED B0 11 08 C2 /...M .6
F020=01 08 00 ED B0 11 20 C2 /...M .6
F028=01 08 00 ED B0 11 30 C2 /...M .06
F030=01 18 00 ED B0 11 98 C2 /...M .6
F038=01 08 00 ED B0 11 80 C3 /...M .u
F040=01 10 00 ED B0 11 A0 C3 /...M .u

```

```

F048=01 30 00 ED B0 11 E8 C3 /...M .Xu
F050=01 08 00 ED B0 11 80 C5 /...M .e
F058=01 10 00 ED B0 11 A8 C6 /...M .4
F060=01 10 00 ED B0 11 C8 C6 /...M .x4
F068=01 08 00 ED B0 11 F0 C6 /...M .P4
F070=01 08 00 ED B0 0B E1 C9 /...M .mAm
F078=26 5A 42 3E 12 22 42 00 /&ZB>."B.
F080=AB 48 3E 08 08 48 30 00 / H>..HO.
F088=6A 56 4A 52 62 42 42 00 /JVJRB8B.
F090=28 10 3A 46 42 42 42 00 /C.FBBB.
F098=8E 42 02 1E 02 02 7E 00 / B....
F0A0=20 10 4A 42 62 5C 40 3C / JBB\<
F0A8=6A 1A 02 1E 02 02 7E 00 /f.....
F0B0=10 28 52 42 42 62 5C 00 /.(RBB\<
F0B8=40 20 18 10 10 10 38 00 /e....8.
F0C0=28 10 7C 02 3C 40 3E 00 /(.I.<@.
F0C8=AC 42 02 3C 40 42 3C 00 / B.<@B<
F0D0=40 20 1C 20 3C 22 5C 00 /e.<".\
F0D8=AE 5A 24 44 44 24 1E 00 / TDDX...
F0E0=68 50 5C 62 42 62 5C 00 /hP\BBB\<
F0E8=58 24 52 42 7E 42 42 00 /XRB\BB.
F0F0=28 10 3A 46 02 02 02 00 /C.F....
F0F8=28 6C 10 10 10 10 10 00 /I.....
F100=20 40 3C 42 7E 02 3C 00 / <B\<

```

```

F108=64 54 44 38 10 10 10 00 /dTD8...
F110=66 58 20 18 04 02 7E 00 /fX....
F118=28 10 7E 20 18 04 7E 00 /(.T....
F120=28 10 3C 42 7E 02 3C 00 /(<B\<
F128=52 6A 52 42 42 42 3C 00 /RJRB8B<
F130=5C 28 08 08 08 08 1C 00 / \.....
F138=58 24 52 42 42 24 18 00 /XRB8B<
F140=20 10 4A 42 42 62 5C 00 / JBB\<
F148=80 40 3C 42 42 42 3C 00 / <BBB<
F150=52 4A 42 42 42 42 3C 00 /RJBB8B<
F158=28 10 3C 42 02 42 3C 00 /(<B.B<
F160=38 54 02 02 02 44 38 00 /8T...D8.
F168=00 00 00 00 00 00 00 00 /.....

```

```

I
I DATABASE I
I R I D S I
I
I V3.2 (4.8.1987) I
I (C) SVEHLA I
I
STISKNI ?

```

```

*

```

```

= P R I K L A D P O U Z I T I =
= D A T A B A Z E R I D S =
=

```

- ZAVEZENÍ V RÁŠTU :

```

READY
LOAD "RIDS-JSS(C)"
FOUND "RIDS-JSS(C)"
LOADING "RIDS-JSS(C)"
READY
RUN

```

```

R I D S
JSS(C)-1987

```

DEFINOVAT ČESKOU ABECEDU ? (A/N)) A
JE NUTNÝ SOUBOR CHY:)ČESKÝ PCG !
↓ PLAY

```

I
I DATABASE I
I R I D S I
I
I V3.2 (4.8.1987) I
I (C) SVEHLA I
I
STISKNI ?

```

*2

PRIKAZ VYZNAM
?.....INFORMACE O PRIKAZECH
R.....STAV DATABASE
C.....POLOZKY DATABASE
+ENJ...POSUV UKAZATELE VPRED
-FNJ...POSUV UKAZATELE VZAD
I.....NALEZENÍ ZAZNAMU
S.....JAKO L SE ZMENOU
E.....ZOBRAZITÍ ÚRSAMU POLOZKY
V.....JAKO E SE ZMENOU
DNJ...ZRUŠENÍ ZAZNAMU
I.....VLOŽENÍ ZAZNAMU PRED BEZHY
B.....PŘIDÁNÍ ZAZNAMU NA KONEC
PNJ...RUZEPSANÍ ZAZNAMU

DALŠÍ PRIKAZY? (A/N)) A

PRIKAZ VYZNAM
N.....PŘIDÁNÍ POLOZKY
N.....PŘEJMENOVÁNÍ POLOZKY
F.....ZRUŠENÍ POLOZKY
U.....VLOŽENÍ DATABASE NA CMT
Z.....ZAVĚZENÍ DATABASE 7 CMT
G.....VYTVOŘENÍ DATABASE
K.....ZRUŠENÍ DATABASE
H.....AKTIVIZACE ZAZNAMU
A.....AKTIVIZACE - OPERACÍ AND
O.....AKTIVIZACE - OPERACÍ OR
X.....STATISTIKA DAT
Y.....SELEKCE TISKU
Q.....KONEC PRÁCE S DATABASE

*G

CEKEJTE PKOSIN !
JMENO NOVE DATABASE.....LIM.
FOURDIE VYTVOŘENÍ (A/N).....A
POČET POLOZEK DATABASE.....3
POČET POLOZEK FORMATU 2.....2

POLOZKA 1
JMENO....PRIJMENI
FORMAT...2

POLOZKA 2
JMENO....JMENO
FORMAT...1

POLOZKA 3
JMENO....TELEFON
FORMAT...1

NEJAKE OPRAVY? (A/N).....A
OPRAVOVANA POLOZKA.....2
ZMENIT JMENO? (A/N).....N
ZMENIT FORMAT? (A/N).....A
NOVY FORMAT.....2

NEJAKE OPRAVY? (A/N).....N

*C

1. PRIJMENI FORMAT = 2
2. JMENO FORMAT = 2
3. TELEFON FORMAT = 1

*R

DATABASE : LIDE
ZAZNAMU : 0
POLOZEK : 3

BEZNY ZAZNAM : 0.

AKTIVNICH ZAZNAMU : 0

*Y

KOLIK ZOBRAZIT POLOZEK.....3
DO RADKU? (A/N).....A

CISLO 1.POLOZKY.....2
CISLO 2.POLOZKY.....1
CISLO 3.POLOZKY.....3

ZOBRAZIT JMENA POLOZEK? (A/N)....A

*I

1. PRIJMENI > KRATKA
2. JMENO > JANA
3. TELEFON > 3271

*B

1. PRIJMENI > IVANIC
2. JMENO > JOSEF
3. TELEFON > 4132

*-

O KOLIK > 3

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JANA | KRATKA | 3271 |

*P

KOLIK > 999

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JANA | KRATKA | 3271 |
| JOSEF | IVANIC | 4132 |

*F

CISLO POLOZKY > 1

OBSAH = IVANIC

*V

CISLO POLOZKY > 3

OBSAH = 4132

NOVY OBSAH > 4232

*B

1. PRIJMENI > MANN
2. JMENO > PETR
3. TELEFON > 1427

*R

DATABAZE : LINE
ZAZNAMU : 3
POLOZEK : 3

BEZNY ZAZNAM : 3.

AKTIVNICH ZAZNAMU : 3

*L

POLUZKA > 2
OBSAH > JANA

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JANA | KRATKA | 3271 |

*S

POLUZKA > 3
OBSAH > 4232

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JOSEF | IVANIC | 4232 |

NOVY OBSAH > 4132

*R

DATABAZE : LINE
ZAZNAMU : 3
POLOZEK : 3

BEZNY ZAZNAM : 2.

AKTIVNICH ZAZNAMU : 3

*D

KOLIK > 1

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 2.

AKTIVNICH ZAZNAMU : 2

*-

O KOLIK > 2

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JANA | KRATKA | 3271 |

*P

KOLIK > 10

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| JANA | KRATKA | 3271 |
| PETR | MANN | 1427 |

*X

POLUZKA > 3

TAKE STANDARDNI ODCHYLKU? (A/N) > A

TELEFON
ZAZNAMU : 2
SOURCE : 4698
PRIMER : 2349

STANDARDNI ODCHYLKA = 922

*A

POLUZKA > 1

<,> NEMU = (KTERA 2 NICH?) > =
HODNOTA > TOHAN

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 2.

AKTIVNICH ZAZNAMU : 0

*O

POLUZKA > 3
<,> NEMU = (KTERA 2 NICH?) > <
HODNOTA > 2000

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 2.

AKTIVNICH ZAZNAMU : 1

*P

KOLIK > 3

| JMENO | PRIJMENI | TELEFON |
|-------|----------|---------|
| PETR | MANN | 1427 |

*H

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 2.

AKTIVNICH ZAZNAMU : 2

*U

CHRANIT ZAPIS? (A/N) > A
HESLO > SHARP

↓ RECORD.PLAY

*Z

EXISTUJICI DATABAZI ZRUSIT? (A/N) > A
JIM-NO DATABAZE 2 CMT > LINE

↓ PLAY
HESLO? > SHARP

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 1.

AKTIVNICH ZAZNAMU : 2

*Y

KOLIK ZOBRAZIT POLOZEK.....3
DO RANKU? (A/N).....N

CISLO 1.POLOZKY.....1
CISLO 2.POLOZKY.....2
CISLO 3.POLOZKY.....3

ZOBRAZIT JMENA POLOZEK? (A/N)....A

*H

FREDCHAZEJICI POLOZKA > 1
JMENO NOVE POLOZKY > VEK
FORMAT NOVE POLOZKY > 1

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 4

BEZNY ZAZNAM : 1.

AKTIVNICH ZAZNAMU : 2

*F

RUSENA POLOZKA > 4

DATABAZE : LINE
ZAZNAMU : 2
POLOZEK : 3

BEZNY ZAZNAM : 1.

AKTIVNICH ZAZNAMU : 2

*N

PREJMEHOVAVANA POLOZKA > 3
NOVE JMENO > JH.

*C

1.PRIJMENI FORMAT = 2
2.VEK FORMAT = 1
3.JH. FORMAT = 2

*P

KOLIK > 3

1.PRIJMENI > KRATKA
2.VEK > 0
3.JH. > JANA

1.PRIJMENI > MANN
2.VEK > 0
3.JH. > PETR

*K

POTVRDITE! (A/N) > A

| | | |
|---|----------|---|
| I | | I |
| 1 | DATABAZE | I |
| I | R I D S | I |
| I | | I |

I V3.2 (4.8.1987) I
I (C) SVEHLA I

STISKNI ?

*Q

READY

SKICÁK

Jan Krejčí, Bouřilova 1104, 198 00 Praha 9

Pro vytvoření programu Skicák mne inspirovalo několik skutečností. Především jsem potřeboval pro některé své programy dokonale graficky provedené obrazovky buď jako úvodní obrazovku, nebo funkční část programu (například mapy, plány a podobně). Dále to byl můj kolega ing. Libeš, který vytvářel obrazovky tak, že si nejprve připravil a dokonale promyslel svůj návrh mimo počítač, například na papíře a ten pak pomocí milimetrového papíru rozložil na jednotlivé body, tak jako je tvořena obrazovka. Tyto body pak přenesl jako jednotlivé bajty do basicového programu (do DATA), vytiskl obrazovku, nahrál ji a tak s ní bylo možné pracovat dál. Tato metoda se mi zdála jako jedna z nejdokonalejších, ale tak pracná, že jsem se rozhodl nějakým způsobem ji ulehčit. Samozřejmě, že nejdokonalejší by bylo rozložit signál z televizní kamery, vytvořit videodigitizér, anebo si jej koupit, či použít jinou mechanickou metodu, nebo dokonalejší programový prostředek. Ale v té době jsem měl k dispozici pouze programy, které pracovaly na obrazovce dosti hrubě a obraz vytvářely pomocí čar a obrázků, nebo i přímým kreslením na obrazovku pomocí světelného pera nebo myši.

Začal jsem proto pracovat na programu, který by využíval shora uvedenou metodu s rozložením předem připraveného návrhu na jednotlivé body a jejich systematickým přenesením do paměti počítače. Pro rozklad předlohy se mi nejlépe osvědčil průhledný milimetrový papír se zdůrazněným rástrem 8 mm, tedy osm bodů. Používám formát A4, který je svými rozměry optimální pro rozklad kreseb, tisků i fotografií. Průsvitný papír přiložím na kresbu, umístím do zvolené polohy a odečítám jednotlivé osmice bodů (v podstatě bajty) ve zdůrazněném čtverečku 8 x 8 bodů a vkládám je pomocí funkce INPUT programu Skicák do paměti Spectra. Postupuji systematicky od levého horního rohu obrázku, po jednotlivých řádcích až k pravému dolnímu rohu.

Po vložení všech částí programu do paměti se program spustí příkazem RUN. To platí i pro případy při přerušení nebo vypnutí z programu.

Hlavní nabídka („menu“)

Spustíme-li program, ocitneme se v hlavní nabídce (menu). Na obrazovce se objeví název programu, číslo jeho verze a seznam funkcí nabídky (v rámečku). Funkcí je celkem pět. Označeny jsou vždy číslem a názvem. K jejich vyvolání je třeba stisknout příslušnou číslici.

1 — EDITOR

Musíme zvolit jednu ze tří pracovních pamětí. Možnost práce se třemi „obra-

zovkami“ oceníme při modifikaci již hotových obrazovek, při jejich míchání a prolínání. Po zvolení čísla pracovní paměti se dostaneme do EDITORu. Ten bude popsán samostatně v následující kapitole.

2 — LIST

Tato funkce je určena pro prohlížení obsahu jednotlivých obrazovek. Stiskem libovolné další klávesy (kromě klávesy „Q“) listujeme jednotlivými obrazovkami, jejichž číslo bliká vždy v pravém horním rohu obrazovky. Návrat do hlavní nabídky je klávesou „Q“.

3 — LOAD

Umožní přečíst data z mgf pásky do jedné ze tří pracovních obrazovek. Zvolená obrazovka se informativně zobrazí. Návrat do hlavní nabídky je automatický.

4 — SAVE

Na mgf pásek bude podle volby zaznamenán obsah jedné ze tří obrazovek (oblasti paměti).

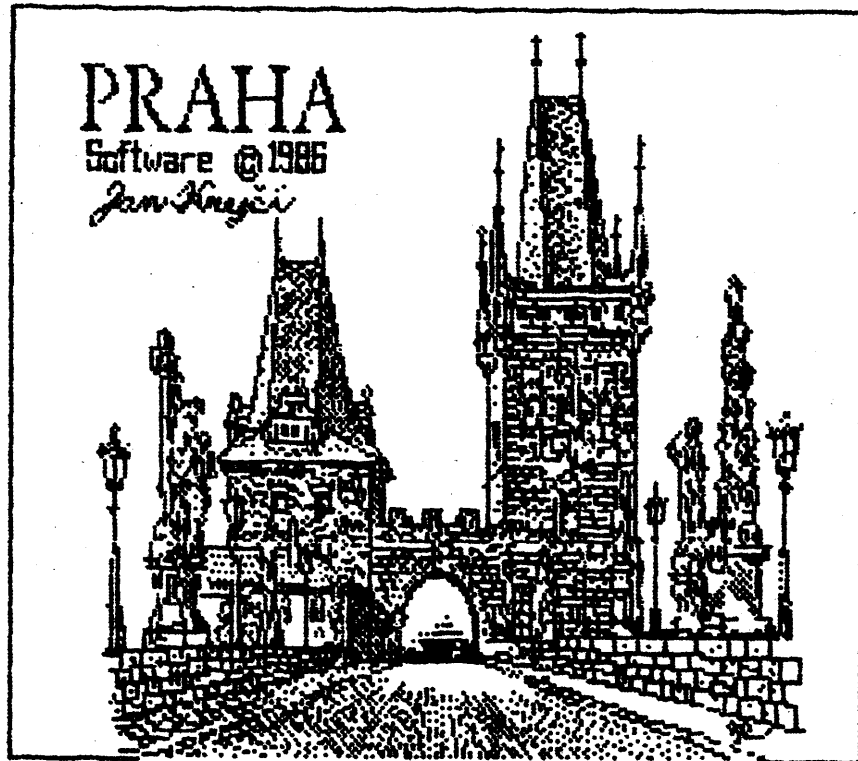
| Obrazovka | Od adresy | Délka |
|-----------|-----------|-------|
| 1 | 40000 | 6912 |
| 2 | 46912 | 6912 |
| 3 | 53824 | 6912 |

Z toho vyplývá, že při nahrání do obrazovky je nutné uložení blíže specifikovat, nebo použít příkazu SCREEN\$. Při vyvolání se zobrazí informativně zvolená obrazovka. Návrat do hlavní nabídky je automatický.

PRAHA

Software © 1986

Jan Krejčí



5 — HELP

Funkce popisuje vlastnosti povelů EDITORu v případě, že ztratíte orientaci při obsluze programu. Postrádá pouze popis vedlejší funkce kurzorových kláves EDITORu. Ta bude popsána v následující části. Návrat do hlavní nabídky je automatický.

Popis EDITORu

Klávesou „1“ z hlavní nabídky a po volbě pracovní obrazovky se dostáváme do EDITORu — nositele myšlenky programu. Slouží k přenášení grafické informace z připravené předlohy a k další úpravě obrazu pomocí několika povelů.

Obrazovka je rozdělena do dvou částí. Ve spodních dvou řádcích (v editační zóně) je nabídka povelů EDITORu a souřadnice kurzoru. Tím můžeme pohybovat pomocí kurzorových kláves 5, 6, 7, 8 po zbývajících, horní části obrazovky. Povel aktivujeme stiskem počátečního inverzně zdůrazněného znaku na klávesnici počítače. Destruktivně působící povel, při kterých dochází k podstatným a nevratným změnám na obrazovce, voláme současným stiskem kláves znaku a kláves CAPS SHIFT nebo SYMBOL SHIFT.

Program umožňuje práci v celém rozsahu obrazovky (192 bodů — 24 řádků). Dva řádky jsou většinou skryty. Objeví se až po najetí kurzoru do zmíněné oblasti. Blikající kurzor vyme-

zuje oblast pro vstup grafických informací — oblast 8 × 8 bodů. Je to v podstatě běžná tisková pozice, kterou dále budeme nazývat znakovka.

Kurzorové klávesy mohou plnit ještě jednu pomocnou funkci. Při stisku kurzorové klávesy (5, 6, 7, 8) zároveň s klávesou SYMBOL SHIFT dojde k překopírování znakovky z místa kurzoru na vedlejší pozici určenou stisknutou kurzorovou klávesou. Tato funkce není v programu nikde uvedena, ani ve funkci HELP hlavní nabídky.

Popis povelů EDITORu

INPUT

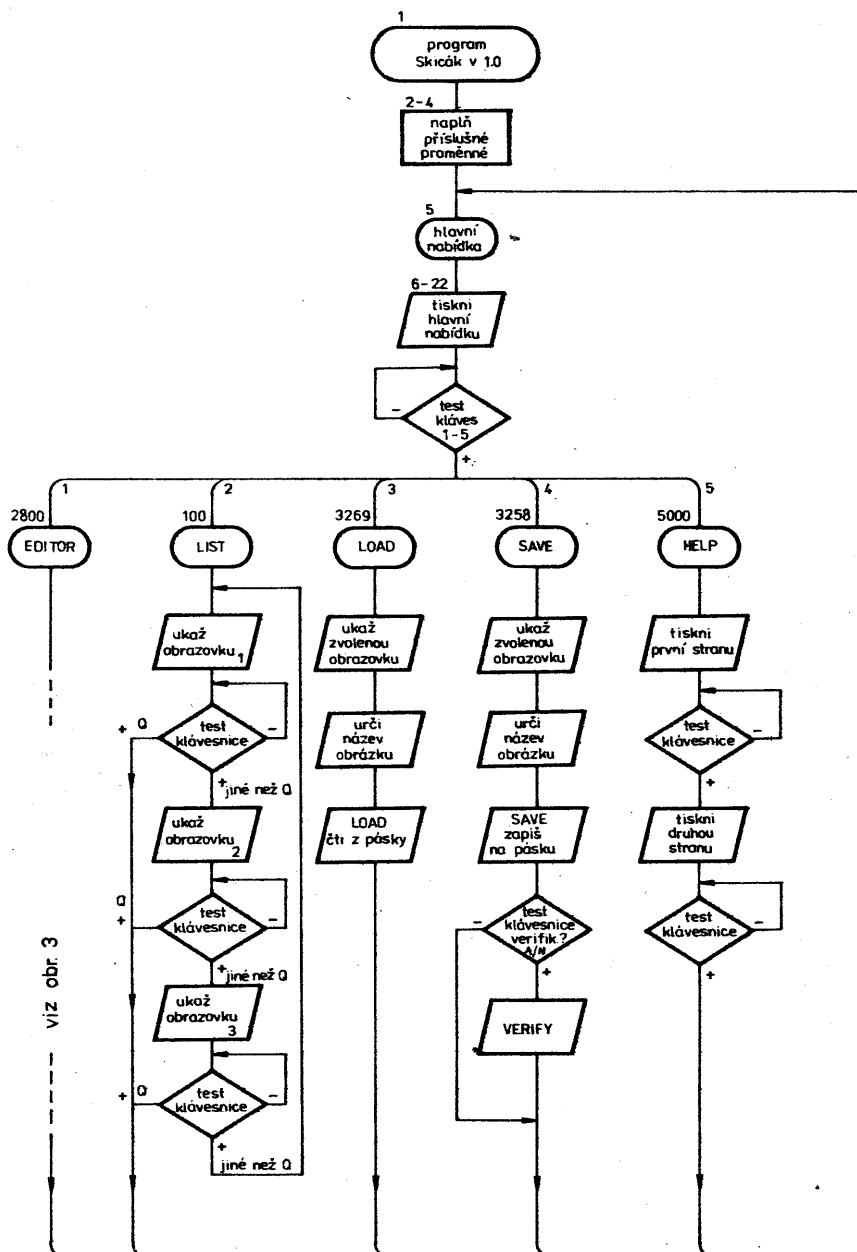
Hlavní povel, kterým přenášíme informace z předlohy několika způsoby.

Stiskem pouhé klávesy „I“ vstoupíme do grafického INPUTu. Uprostřed obrazovky se nám zobrazí osmkrát zvětšená znakovka, na které je kurzor. Oblasti PAPER jsou vyznačeny znakem „O“ a oblasti INK tmavou plnou znakovkou. Při invertovaném obrazu je to obráceně. V editační zóně se objeví nová nabídka s komentářem. Kurzorem ve zvětšené znakovce můžeme pohybovat pomocí kurzorových kláves. Klávesou „O“ měníme pod kurzorem oblast z INK na PAPER a obráceně. Povel CLS plní dvě funkce, prostým stiskem „C“ se obnoví původní grafická informace ve znakovce, nebo se současným stiskem CAPS SHIFT se celá znakovka vymaže.

Povelem „Q“ — QUIT opouštíme zadávání na dané pozici a vracíme se buď do editoru, nebo pokračujeme v plnění znakovky vpravo od současné pozice, či na dalším řádku, pokud jsme plnili poslední tiskovou pozici na předchozím řádku.

Druhou variantu povelu INPUT vyvoláme stiskem „I“ současně s CAPS SHIFT. V tomto případě budeme každou tiskovou pozici, na které je kurzor, plnit po jednotlivých bajtech — osmkrát zadáme hodnotu. Při vkládání jednotlivých hodnot je zobrazena také nabídka variant vkládání. Můžeme totiž hodnoty zadávat ve třech číselných soustavách, nebo vytisknout na pozici kurzoru znak, nebo celou pozici vyplnit jednou hodnotou. Máme také možnost se vrátit k zadávání od prvního bajtu. Při vložení prázdného řetězce ukončeném ENTER vynecháme jeden bajt. Ten zůstává nezměněn. Takovýmto způsobem je možné měnit ve znakovce pouze jeden nebo několik bajtů. Způsob zadávání určíme prvním znakem ve vkládaném řetězci. Nabídka těchto znaků je v prvním editačním řádku. Návrat z tohoto typu povelu je automatický.

Podobným způsobem, jako tyto dvě varianty povelu INPUT, pracují i programy pro vytváření nových znakových souborů (ASCII, UDG), například TYPOGRAPH nebo 4D UDG a pod. Nic nám nebrání použít program SKICÁK i tímto způsobem, stačí si uvědomit, kde jsou v paměti uloženy jednotlivé obrazovky. Tam by se ukládaly i nové vytvořené znakové soubory. Po vytvoření by je bylo nutné přemístit na místa v paměti podle našeho záměru a změnit příslušné systémové proměnné, pro ASCII proměnnou CHARS — 23606/7 a pro UDG proměnnou UDG — 23675/6.



MEM

Funkce usnadňuje přenos a „množení“ stejných znakových pozic na obrazovce. Stiskem klávesy „M“ se současným CAPS SHIFT zaznamenáme znakovku pod kurzorem do mezipaměti. Vyvolat na pozici kurzoru ji pak můžeme kolikrát chceme stiskem kláves „M“ a SYMBOL SHIFT.

Další funkce pracují s celou obrazovkou a jsou určeny pro úpravu větších částí nebo celých obrazovek.

BARVA

Vyvoláním aktivujeme novou nabídku, která nám nabízí tři možnosti. Změnit INK nebo PAPER v celém obraze, nebo celou obrazovku invertovat — INV. Návrat do EDITORu je automatický.

COPY

Provede kopii obrazovky na tiskárně (ZX PRINT a jeho ekvivalenty). Kopíruje však všech 24 řádek.

DEL

Funkce DELETE je z nejdestruktivnějších. Má dvě podoby. První, vyvolaná stiskem kláves „D“ současně s CAPS SHIFT, maže oblast Display file, tj. oblast paměti, která určuje, kde je INK a kde PAPER (prvních 6144 bajtů obrazovky). Druhá, aktivovaná stiskem kláves „D“ a „SYMBOL SHIFT“, maže oblast atributů, tj. oblast, která určuje barvy a další vlastnosti znakových pozic (posledních 768 bajtů obrazovky).

EXCH — EXCHANGE

Povel, který umožňuje změnu pracovní obrazovky.

NEW

Nejdestruktivnější, stiskem klávesy „N“ současně s CAPS SHIFT se maže celá pracovní obrazovka.

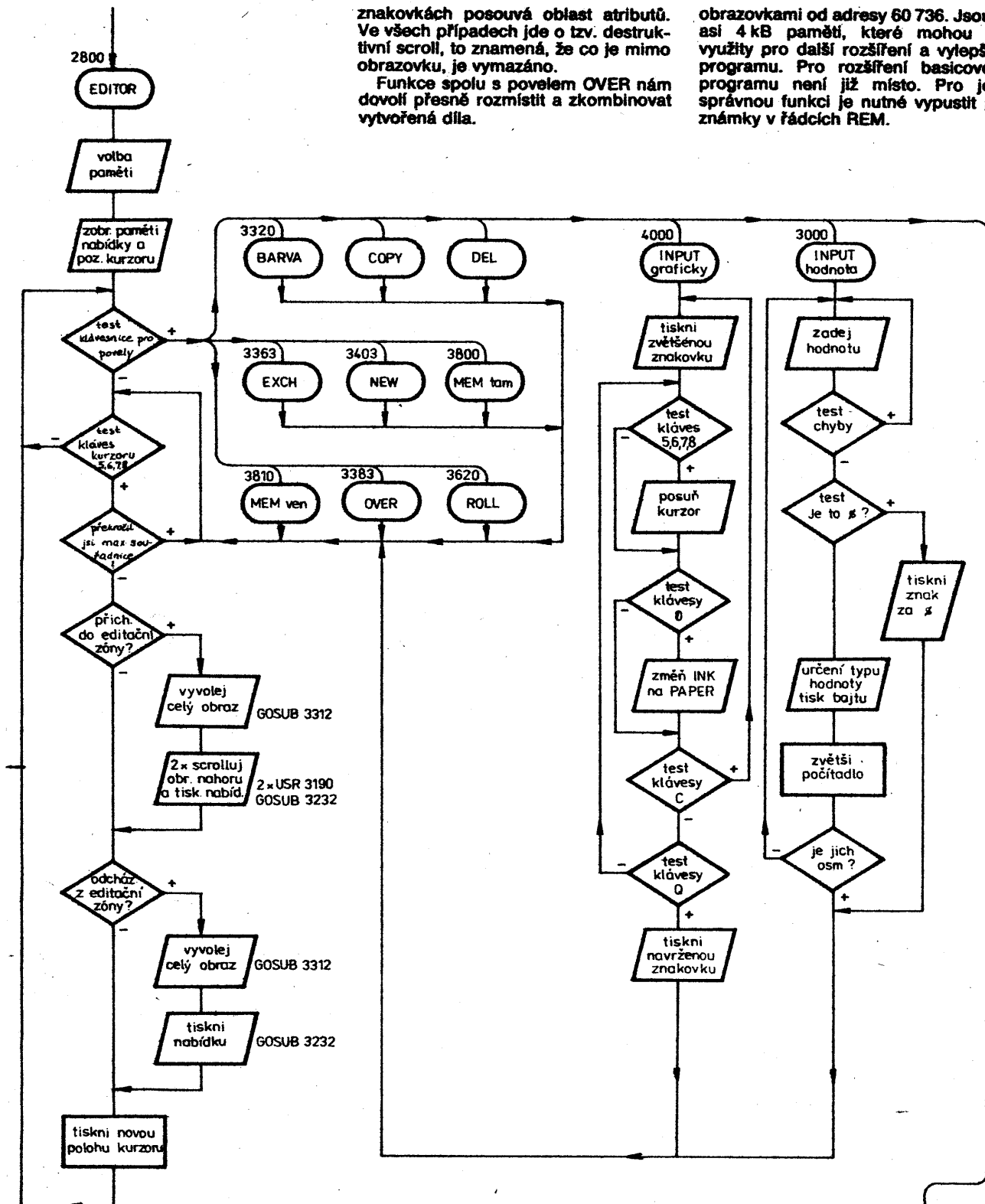
OVER

Tato funkce smíchá obsahy dvou obrazovek — pracovní se zvolenou.

znakovkách posouvá oblast atributů. Ve všech případech jde o tzv. destruktivní scroll, to znamená, že co je mimo obrazovku, je vymazáno.

Funkce spolu s povelem OVER nám dovolí přesně rozmístit a zkombinovat vytvořená díla.

obrazovkami od adresy 60 736. Jsou to asi 4 kB paměti, které mohou být využity pro další rozšíření a vylepšení programu. Pro rozšíření basicového programu není již místo. Pro jeho správnou funkci je nutné vypustit poznámky v řádcích REM.



QUIT

Funkce návratu do hlavní nabídky.

ROLL

Po aktivaci stiskem klávesy „R“ můžeme pomocí kurzorových kláves pohybovat obrazovkou do všech směrů a to třemi způsoby. Samotným stiskem některé kurzorové klávesy se posouvá (scrolluje) display file po jednom bodu. Se současným stiskem CAPS SHIFT se posouvá display file daným směrem, ale po jednom znaku (osmi bodech). Při stisku se SYMBOL SHIFT se po

Popis struktury a funkce programu

Uložení v paměti

Program Skicák je v paměti počítače uložen ve třech částech — basicový program, strojové rutiny a paměť obrazovek, vlastně data programu. Uložení jednotlivých částí je patrné z obr. 1. Podstatnou část paměti zabírá Basicový program, podporovaný ve své funkci rutinami ve strojovém kódu, které jsou umístěny na konci paměti před UDG. Obrazovky jsou uloženy těsně za basicovým programem asi uprostřed paměti RAM. Jediná volná paměť je za

Struktura a funkce programu

Funkční diagramy programu Skicák jsou na str. 62, kde je část hlavní nabídky a na str. 63, kde je popsána

stručně funkce EDITORu. V diagramech jsou orientačně uvedena čísla řádků. Program Skicák nahrajeme s autostartem od řádky 6000 a strojový kód uložíme na pásku za něj. V řádcích od 7000 je v programu uložena informace o rozmištění českých znaků v UDG.

Struktura a funkce EDITORu

EDITOR — funkce „1“ hlavní nabídky — je uložen mezi řádky 2800 a 5000. Jeho grafické schéma je na str. 63. Jedná se o podobnou strukturu, jako v případě hlavní nabídky, je zde ale i nejsložitější problém programu — pohyb kurzoru po obrazovce a vyhodnocení jeho pozic v případě práce na nejnižších dvou řádcích obrazovky. To vše má za úkol podprogram od řádku 3408, který je opakovaně volán po testu klávesnice na volaný povel. Další, vnořené podprogramy tohoto podprogramu, jsou využívány také některými funkcemi editoru (INPUT a MEM).

Popis použitých strojových rutin, rutin v ROM a neobvyklých vlastností BASICu.

Rutiny strojového kódu začínají od adresy 64 599. První je rutina COPY — 24 řádků, využívá podprogramu ROM 3762. Další rutina užívá povel ROLL pro posuv display file o znak dolů — 64609. Rutina 64682 posouvá obě části obrazovky o znak dolů. Následující sedm rutin je použito ze známého programu SUPERCODE. První dvě (64744 a 64756) slouží pro přenesení obsahu viditelné obrazovky do pracovní paměti a naopak. Tyto rutiny využívají skoro všechny povely a funkce programu. Rutiny 64784 a 64828 nejsou využity, zůstaly v paměti po úpravách předchozích pracovních verzí. První vymění obsahy obrazovky a pracovní paměti a druhá naplní určený obdélník v obrazovce jedním znakem. Následující tři podprogramy používá funkce BARVA. 64858 mění INK v celém obraze, 64883 mění PAPER v celém obraze a 64809 provede inverzi obrazovky. Pak následuje rutina funkce OVER — 64914 a za ní rutiny pro funkci ROLL. 64935 posune Display file o znak vlevo, 64956 vpravo, 64978 posune Display file o bod vlevo, 64995 o bod vpravo, 65012 o bod nahoru a 65168 o bod dolů. Posuv atributů řídí rutiny — 65103 vlevo, 65125 vpravo a 65147 nahoru.

Další rutiny jsou pro mazání jednotlivých oblastí obrazovky — 65326 maže atributy a 65342 maže Display file. Od adresy 65368 začíná uživatelská grafika se znaky české abecedy (viz řádek 7000).

Program používá ve své funkci nejen těchto rutin, ale i některé programy ROM a změny systémových proměnných, ale také v manuálu nepopsané funkce Basicu pro tisk v editační zóně (řádek 3236). Pro mazání editační zóny volá rutinu 3438, pro změnu barvy editační zóny mění systémovou proměnnou BORDCR — 23624, pro změnu počtu editačních řádků mění systémovou proměnnou DF_SZ — 23659 a pro

scroll obrazovky nahoru o řádek volá rutiny 3190 nebo 3582.

Náměty pro další vylepšení a rozšíření programu Skicák

Každý program, tak jako jiná díla lidského intelektu a rukou, je možné dotvořit — zlepšit, zjednodušit a „polidštit“. Program SKICÁK v 1.0 není v tomto ohledu žádná výjimka, zdá se mi, že spíše naopak. Mým prapůvodním cílem bylo dovést program k jakés takés dokonalosti vytvořením grafického editoru typu GEM a ovládat jej joystickem, myší nebo světelným perem. Ovšem člověk mění a věci se mění. Objevil se program ART STUDIO a tak jsem skromně ustoupil a zjednodušil svůj program tak, aby byl i vedle tohoto „velikána“ užitečný a alespoň částečně doplnil některé jeho funkce. Rád bych však v této části upozornil na jisté slabiny a rezervy programu. V první řadě postrádá místo pro rozšíření BASICu. To by se dalo řešit buď posunutím obrazovek směrem ke konci paměti, anebo využitím prázdného místa nad nimi a převedením některé části BASICu do strojového kódu, například řízení kurzoru. BASIC také můžeme mírně „zahustit“ úpravou jeho struktury — vypuštěním poznámek, nebo vytvořením podprogramů při několikerém opakování operací (viz řádky 114, 116, 118 — test klávesnice). Také zmenšení počtu řádků a využití funkce VAL pro ukládání celočíselných

proměnných by uvolnilo nějaký prostor. Ve funkci se dá program zdokonalit především v práci s barvami. Zvolený tisk kurzoru (transparentní obarvení INK8, PAPER8 na řádcích 3432 až 3448) se projeví zejména při zpracovávání obrazovek s mnohobarevnými, blikajícími a zdůrazněnými plochami; kurzor nám bude na viditelné obrazovce v těchto místech rušit FLASH a BRIGHT. Při nejbližším dalším vyvolání obrazu z paměti se však obraz obnoví v původním stavu. Zde by bylo vhodné program zdokonalit tak, aby testoval hodnoty barev před umístěním kurzoru a navracel je do původního stavu po jeho dalším pohybu. Rovněž by stálo za to doplnit program vhodnou rutinou pro obsluhu chyb a přerušení, té by se dalo využít i při návratu z jednotlivých funkcí a do hlavní nabídky. Odpadly by tak testy klávesy „Q“.

*** S k i c á k V1.0 ***

- 1 - EDITOR
- 2 - LIST
- 3 - LOAD
- 4 - SAVE
- 5 - HELP

*** Jan Krejčí ***

Začátek Basicového programu

| | | |
|------------------|------------|---|
| BASIC | | podle systémové proměnné PROG Bez připojeného Interface I — 23755 |
| | | konec Basicového programu, asi 39,5k |
| Volno | (asi 0,5k) | |
| Obrazov- ka 1 | | RAMTOP = 39999/začátek první obrazovky 40000 |
| | | začátek druhé obrazovky 46912 |
| | | začátek třetí obrazovky 53824 |
| Volná paměť | | začátek volné paměti |
| Stroj- rutiny | | začátek strojových rutin 64599 |
| UDG | | začátek UDG — 65368 |

konec paměti 65536

Typis programu SKICAK

```

1 REM * PROGRAM SKICAK *
2 POKE 23624,56: CLS
3 DIM w(8): LET o2=156: LET R
am=40000
4 LET x1=0: LET y1=0: LET zve
=0: LET rop=0: LET sk=0: LET s$=
": LET o1=64: LET por=0: LET x=
0: LET y=0: LET kurz=22528+x+y+3
2: LET atr=ATTR (x,y)
5 REM *menu*
6 CLS : PRINT AT 7,10;"1 - ED
ITOR"
15 PRINT AT 9,10;"2 - LIST"
16 PRINT AT 11,10;"3 - LOAD"
17 PRINT AT 13,10;"4 - SAVE"
18 PRINT AT 2,4;"*** S k i c a
k U1.0 ***"
19 PRINT #1;AT 0,12;"© 1986"
20 PRINT AT 20,6;"*** Jan Krej
ci ***"
21 PRINT AT 15,10;"5 - HELP"
22 PLOT 27,27: DRAW 0,115: DRA
U 199,0: DRAW 0,-115: DRAW -199,
0
30 IF INKEY$="1" THEN GO TO 28
00
32 IF INKEY$="3" THEN GO SUB 3
268: GO TO 5
34 IF INKEY$="4" THEN GO SUB 3
256: GO TO 6
36 IF INKEY$="2" THEN GO TO 10
0
38 IF INKEY$="5" THEN GO TO 50
00
40 GO TO 30
100 REM LIST
114 LET o2=156: LET Ram=40000:
GO SUB 3312: PRINT FLASH 1; BRIG
HT 1;AT 4,28;"1": PAUSE 0: IF IN
KEY$="q" OR INKEY$="0" THEN GO T
O 2
116 LET o2=183: LET Ram=46912:
GO SUB 3312: PRINT FLASH 1; BRIG
HT 1;AT 4,28;"2": PAUSE 0: IF IN
KEY$="q" OR INKEY$="0" THEN GO T
O 2
118 LET o2=210: LET Ram=53824:
GO SUB 3312: PRINT FLASH 1; BRIG
HT 1;AT 4,28;"3": PAUSE 0: IF IN
KEY$="q" OR INKEY$="0" THEN GO T
O 2
120 GO TO 114
2800 REM EDITOR
2810 CLS : GO SUB 3364
2820 GO SUB 3232
2830 GO SUB 3600
2840 IF INKEY$="I" THEN GO TO 30
00: LET x=x+1
2844 IF INKEY$="r" THEN GO SUB 3
620: GO TO 2820
2846 IF INKEY$="i" THEN GO TO 40
00: LET x=x+1
2850 IF INKEY$="o" THEN GO SUB 3
384
2855 IF INKEY$="c" THEN GO SUB 3
312: RANDOMIZE USR 64599: GO SUB
3308: GO SUB 3232
2860 IF INKEY$="e" THEN GO SUB 3
364: GO SUB 3232
2864 IF INKEY$="h" THEN GO SUB 3
800
2868 IF INKEY$="." THEN GO SUB 3
810
2870 IF INKEY$="b" THEN GO SUB 3
320
2880 IF INKEY$="n" THEN GO SUB 3
404
2882 IF INKEY$="CHR$ 205 THEN GO
SUB 3312: RANDOMIZE USR 65326: G
O SUB 3308: GO SUB 3232
2884 IF INKEY$="d" THEN GO SUB 3
312: RANDOMIZE USR 65342: GO SUB
3308: GO SUB 3232
2890 IF INKEY$="q" THEN GO TO 1
2900 IF INKEY$="(" THEN LET kam=
1: GO SUB 3524: GO SUB 3460: GO
SUB 3500
2910 IF INKEY$="X" THEN LET kam=
-1: GO SUB 3532: GO SUB 3460: GO
SUB 3500
2920 IF INKEY$="&" THEN LET kam=
32: GO SUB 3460: GO SUB 3460: GO
SUB 3540: GO SUB 3500: IF zve=1
THEN LET y=21: LET rop=0: GO SU
B 3312: GO SUB 3232: LET zve=0
2930 IF INKEY$="." THEN GO SUB 3
492: LET kam=32: GO SUB 3460: G
O SUB 3516: GO SUB 3500
2940 GO SUB 3408: GO TO 2830
3000 REM Input
3002 GO SUB 3460
3004 FOR i=0 TO 1792 STEP 256
3008 BEEP 0.05,i/256
3012 INPUT AT 0,2;"Naplněte hex,
in;znaky;řádky;AT 1,2;"Zadávaš
byte č. ";(i/256+1);AT 1,21: LIN
E $
3016 IF s$="s" OR s$="S" THEN LE
T i=0: GO TO 3000
3020 IF s$=" " THEN GO TO 3060
3024 IF s$(1)="z" OR s$(1)="Z" T
HEN GO SUB 3116: GO TO 3064
3028 IF s$(1)="#" THEN GO SUB 32
88: GO TO 3052
3032 IF s$(1)="$" THEN PRINT AT
y+1 AND por=1)+(2 AND por=2),x;
s$(2): LET kam=0: GO SUB 3460: G
O SUB 3500: GO TO 3064
3036 IF s$(1)="B" OR s$(1)="b" T
HEN GO SUB 3200: GO TO 3052
3040 IF CODE s$=58 OR CODE s$=
47 THEN PRINT #1;AT 0,1;"Spaťné
zapsaný znak !!!": BEEP 0.8,20:
GO TO 3012
3044 IF VAL s$>255 THEN PRINT #1

```

```

;AT 0,1;"Prilis velké číslo !!!"
: BEEP 0.2,24: BEEP 0.4,20: GO T
O 3012
3048 LET wej=VAL s$
3052 IF wej>255 THEN PRINT #1;AT
0,1;"Prilis velké číslo !!!": B
EEP 0.8,20: GO TO 3012
3056 POKE 16384+abc+x+(y+por-yy)
*32+i,wej: POKE Ram+abc+x+(y+por
+rop-yy)*32+i,wej
3060 NEXT i
3064 FOR u=0 TO 8: BEEP 0.02,4+u
: NEXT u: PAUSE 0.5
3068 PRINT #1;AT 0,1;"Pozice kur
zoru x=";x;" y=";y;" y+por
3072 PRINT #1;AT 1,1;"Chcete pokr
ačovat v zápisu A/N ?"
3076 IF INKEY$="n" THEN RANDOMIZ
E USR 3438: BEEP 0.08,3: BEEP 0.
09,10: GO TO 2820
3080 IF INKEY$="a" THEN GO SUB 3
100: GO TO 3092
3084 GO TO 3072
3088 LET x=x+1
3092 FOR w=8 TO 0 STEP -1: BEEP
0.01,RND*10-w: NEXT w
3096 PRINT BRIGHT 1; FLASH 1; OV
ER 1; PAPER 8; INK 8;AT y+sk+por
,x;: PRINT OVER 1; INK 8; PAP
ER 8;AT y+sk+por-(1 AND x=0)-(21
AND x=31 AND y=21),x-1+(32 AND
x=0)-(21 AND x=31 AND y=21);:
GO TO 3004
3100 IF x=31 AND por=2 THEN LET
x=0: LET y=0: LET rop=0: LET por
=0: LET abc=0: LET yy=0: GO SUB
3312: GO SUB 3232: RETURN
3104 IF x=31 AND y+rop=21 THEN
LET x=0: GO SUB 3284: RETURN
3108 IF x=31 AND y+rop<21 THEN L
ET x=0: LET y=y+1: RETURN
3112 GO TO 3088
3116 IF s$(2 TO )="" THEN RETURN
3118 IF VAL s$(2 TO )>255 THEN R
ETURN
3120 FOR e=0 TO 1792 STEP 256
3124 POKE 16384+abc+x+(y+por-yy)
*32+e,VAL s$(2 TO ): POKE Ram+ab
c+x+(y+por+rop-yy)*32+e,VAL s$(2
TO )
3128 NEXT e: RETURN
3200 LET wej=VAL ("BIN "+s$(2 TO
LEN s$))
3204 PRINT #1;AT 0,1;"Decimálně
wej: PAUSE 20: RETURN
3208 LET wej=0: LET kruh=0
3212 FOR w=LEN s$ TO 2 STEP -1
3216 LET wej=wej+((CODE s$(w)-48
-7*(CODE s$(w)>65))*16^kruh)
3220 LET kruh=kruh+1
3224 NEXT w
3228 RETURN
3232 POKE 23624,112: RANDOMIZE U
SR 3438
3236 PRINT #1;AT 0,0;"BARVA:ROP
Y:EL:BXCH:INPUT:NEW:"
3238 PRINT #1;AT 1,0;"SEM:OVER:
QUIT:BOLL:"
3240 RETURN
3244 IF por=0 THEN RETURN
3248 BEEP 0.01,32: IF por=1 THEN
LET smer=0: LET por=0: LET y=21
: LET rop=0: GO SUB 3312: GO SUB
3232: GO TO 2830
3252 IF por=2 THEN LET por=1: LE
T smer=3: RETURN
3258 CLS : INPUT "název obrázku
LINE $
3260 GO SUB 3364: POKE 23736,187
: SAVE z$CODE Ram,6912
3261 CLS : PRINT AT 10,3;"Chcete
zkontrolovat A/N ?"
3262 IF INKEY$="n" THEN GO TO 32
65
3263 IF INKEY$="a" OR INKEY$="A"
THEN GO SUB 3312: VERIFY z$CODE
Ram: CLS : PRINT AT 10,10;"Dobr
ý !!!": BEEP 0.2,33: PAUSE 4: GO
TO 3265
3264 GO TO 3262
3266 RETURN
3269 CLS : GO SUB 3364: INPUT "n
ázev nahrávky; LINE x$
3270 CLS : PRINT AT 10,6;"Nahráv
ka se obraceá
3272 LOAD x$SCREEN$
3276 GO SUB 3308
3280 RETURN
3284 IF y<21 AND por=0 THEN RETU
RN
3288 IF por=0 THEN LET por=1: GO
SUB 3548: GO TO 3296
3292 IF por=1 THEN LET por=2
3296 LET y=19: LET rop=2: PRINT
#1;AT 1,26;"y=";y+por+rop;:
3300 BEEP 0.01,30: LET smer=2
3304 RETURN
3307 REM zaznam
3308 POKE 64745,01: POKE 64746,0
2: RANDOMIZE USR 64744: RETURN
3310 REM ukaz
3312 POKE 64757,01: POKE 64758,0
2: RANDOMIZE USR 64756: RETURN
3315 REM vyme
3316 POKE 64785,01: POKE 64786,0
2: RANDOMIZE USR 64784: RETURN
3318 REM BARVA
3320 RANDOMIZE USR 3438: PRINT #
1;AT 0,1;"Zvolte stiskem čísla"
: PRINT #1;AT 1,1;"1 = INV; 2 =
INK; 3 = PAPER"
3324 IF INKEY$="1" THEN GO TO 33
48
3328 IF INKEY$="2" THEN GO TO 33

```

```

40
3332 IF INKEY$="3" THEN GO TO 33
44
3336 GO TO 3324
3340 INPUT "číslo inkoustu";ink
: POKE 64859,ink: GO SUB 3312: R
ANDOMIZE USR 64858: GO SUB 3308:
GO SUB 4860: GO TO 2820
3344 INPUT "číslo papíru";pap:
POKE 64884,pap: GO SUB 3312: RAN
DOMIZE USR 64883: GO SUB 3308: G
O SUB 4860: GO TO 2820
3348 GO SUB 3312: RANDOMIZE USR
64809: GO SUB 3308: GO SUB 4860:
GO TO 2820
3352 IF por=1 THEN RANDOMIZE USR
3190: PRINT OVER 1;AT 21,0;:
3356 IF por=2 THEN RANDOMIZE USR
3190: PRINT OVER 1;AT 21,0;:
RANDOMIZE USR 3190: PRINT OVER 1
;AT 21,0;:
3360 GO SUB 3232: RETURN
3363 REM BXCH
3364 INPUT "číslo paměti?(1-3) "
:Rom
3368 IF Rom=1 THEN LET o2=156: L
ET Ram=40000
3372 IF Rom=2 THEN LET o2=183: L
ET Ram=46912
3376 IF Rom=3 THEN LET o2=210: L
ET Ram=53824
3380 GO SUB 3312: GO SUB 3352: R
ETURN
3383 REM OVER
3384 INPUT "číslo paměti(1-3) -
mer: POKE 64919,64
3388 IF mer=1 THEN POKE 64920,15
6
3392 IF mer=2 THEN POKE 64920,18
3
3396 IF mer=3 THEN POKE 64920,21
0
3400 GO SUB 3312: RANDOMIZE USR
64915: GO SUB 3308: GO SUB 3352:
GO SUB 3232: RETURN
3403 REM NEW
3404 POKE 23624,atr: CLS : GO SU
B 3308: GO SUB 3232: RETURN
3408 REM řizení kurzoru
3409 LET nast=1
3412 LET smer=0
3416 IF INKEY$="5" THEN BEEP 0.0
03,x+10: IF x>0 THEN LET smer=1:
LET x=x-1: LET nast=1: GO TO 34
32
3420 IF INKEY$="6" THEN GO SUB 3
284: BEEP 0.003,y+10: IF y<21 AN
D por=0 THEN LET smer=2: LET yy
+1: LET nast=1: GO TO 3432
3424 IF INKEY$="7" THEN GO SUB 3
244: BEEP 0.003,y+10: IF y>0 AN
D por=0 THEN LET y=y-1: LET smer=
3: LET nast=1: GO TO 3432
3428 IF INKEY$="8" THEN BEEP 0.0
03,x+10: IF x<31 THEN LET smer=4
: LET x=x+1: LET nast=1: GO TO 3
432
3432 IF nast=1 THEN PRINT BRIGHT
1; FLASH 1; OVER 1; PAPER 8; IN
K 8;AT y+sk+por,x;:
3436 IF smer=1 THEN PRINT OVER 1
; PAPER 8; INK 8;AT y+sk+por,x+1
;:
3440 IF smer=2 THEN PRINT OVER 1
; PAPER 8; INK 8;AT y+por-1+sk,x
;:
3444 IF smer=3 THEN PRINT OVER 1
; PAPER 8; INK 8;AT y+1+por+sk,x
;:
3448 IF smer=4 THEN PRINT OVER 1
; PAPER 8; INK 8;AT y+por+sk,x-1
;:
3452 LET sk=0: LET nast=0
3456 RETURN
3460 LET cba=0: LET yyy=0
3464 IF y<7 THEN LET abc=0: LET
yyy=0: RETURN
3468 IF y>7 AND y<15 THEN LET a
bc=2048: LET yy=6: RETURN
3472 IF y>15 AND y<23 THEN LET
abc=4096: LET yy=16: RETURN
3476 RETURN
3480 IF por=2 THEN BEEP 0.02,25:
BEEP 0.03,30: GO TO 2830
3484 IF y=7 OR y=15 THEN LET kam
=0: LET cba=2048: LET yyy=7: RET
URN
3488 RETURN
3492 IF y=0 THEN BEEP 0.02,23: B
EEP 0.03,33: GO TO 2940
3496 RETURN
3500 LET vpr=0: FOR i=0 TO 1792
STEP 256: BEEP 0.002,30
3504 LET vpr=PEEK (16384+abc+x+(
y+por-yy)*32+i)
3508 POKE 16384+abc+cba+x+kam+(y
+por-yy-yyy)*32+i,vpr: POKE Ram+
abc+cba+x+kam+(y+por+rop-yy-yyy)
*32+i,vpr
3512 NEXT i: RETURN
3516 IF y=8 OR y=16 THEN LET kam
=0: LET cba=2048: LET yyy=7
3520 RETURN
3524 IF x=31 AND por=2 THEN BEEP
0.01,30: BEEP 0.01,34: GO TO 29
40

```

```

3523 RETURN
3532 IF x=0 THEN BEEP 0.01,35: B
EEP 0.02,35: GO TO 2940
3536 RETURN
3540 IF y=21 AND por=0 THEN GO 5
UB 3548: LET rop=2: LET y=19: LE
T s=2: LET n=1: GO SUB 343
2: LET zve=1: RETURN
3544 RETURN
3548 POKE 23624,56: CLS: GO SUB
3512: POKE 23659,0: RANDOMIZE U
SR 3582: RANDOMIZE USR 3582: POK
E 23659,2: POKE 23624,112: GO SU
B 3232: RETURN
3600 PRINT #1:AT 1,21;"X=";X;" "
3602 PRINT #1:AT 1,26;"Y=";Y+por
+rop;
3604 RETURN
3620 REM BOLL
3622 GO SUB 3312: POKE 23624,56:
GO SUB 3640: GO SUB 3308
3624 IF por=1 OR por=2 THEN RAND
OMIZE USR 3190: RANDOMIZE USR 31
90
3628 RETURN
3642 LET a$=INKEY$
3644 LET n=(65103 AND a$="X")+16
5125 AND a$="(")+(65147 AND a$="
")+(64682 AND a$="&")+165258 AN
D a$=CHR$(11)+(64609 AND a$=CHR$
10)+(64935 AND a$=CHR$(8)+(6495
6 AND a$=CHR$(9)+(64978 AND a$="
5")+(64995 AND a$="8")+(65012 AN
D a$="7")+(65168 AND a$="6")
3646 IF a$="0" THEN GO SUB 3312:
GO TO 2820
3648 IF a$="9" THEN RETURN
3650 IF n<0 THEN BEEP 0.001,30:
RANDOMIZE USR n
3652 GO TO 3640
3656 REM REM tsm
3658 GO SUB 3464: LET sv=1
3660 FOR i=0 TO 1792 STEP 256: B
EEP 0.002,30
3664 LET sv=PEEK (16384+abc+x
+(y+por-yy)*32+i): LET sv=sv+1
3666 NEXT i: RETURN
3670 REM REM ven
3672 GO SUB 3460: LET sv=1
3674 FOR i=0 TO 1792 STEP 256: B
EEP 0.002,30
3678 POKE 16384+abc+cba+x+(y+por
-yy-yy)*32+i,(sv): POKE Ram+ab
c+cba+x+(y+por+rop-yy-yy)*32+i,
v(sv)
3682 LET sv=sv+1
3684 NEXT i: RETURN
4000 REM INPUT
4002 BEEP 0.03,40: BEEP 0.04,20
4004 LET xz=1: LET yz=1: LET a=5
LET b=11
4006 POKE 23624,112: RANDOMIZE U
SR 4438: PRINT #1:AT 1,6;"S:SU
IT:"
4008 PRINT #1:AT 0,2;"Ovládání k
lávesek 5 6 7 8 0"
4010 GO SUB 3600
4012 DIM f$(8,8)
4014 PLOT 95,63: DRAW 0,65: DRAW
65,0: DRAW 0,-65: DRAW -65,0
4016 PLOT 94,62: DRAW 0,67: DRAW
67,0: DRAW 0,-67: DRAW -67,0
4018 PLOT 93,61: DRAW 0,69: DRAW
69,0: DRAW 0,-69: DRAW -69,0
4020 PLOT OVER 0,95,63: DRAW OVE
R 1,0,65: DRAW OVER 1,65,0: DRAW
OVER 1,0,-65: DRAW OVER 1,-65,0
4022 FOR i=1 TO 8: PRINT AT a+i,
b+1: PAPER 5: BRIGHT 1;"
NEXT i
4024 GO SUB 3464: LET i=1
4026 FOR j=0 TO 1792 STEP 256
4028 LET p=PEEK (Ram+abc+x+(y+p
or+rop-yy)*32+j)
4030 LET f$(i,1)=STR$(pp-128)=0
4032 IF pp-128=0 THEN LET pp=pp
-128
4034 LET f$(i,2)=STR$(pp-64)=0
4036 IF pp-64=0 THEN LET pp=pp-
64
4038 LET f$(i,3)=STR$(pp-32)=0
4040 IF pp-32=0 THEN LET pp=pp-
32
4042 LET f$(i,4)=STR$(pp-16)=0
4044 IF pp-16=0 THEN LET pp=pp-
16
4046 LET f$(i,5)=STR$(pp-8)=0
4048 IF pp-8=0 THEN LET pp=pp-8
4050 LET f$(i,6)=STR$(pp-4)=0
4052 IF pp-4=0 THEN LET pp=pp-4
4054 LET f$(i,7)=STR$(pp-2)=0
4056 IF pp-2=0 THEN LET pp=pp-2
4058 LET f$(i,8)=STR$(pp-1)=0
4060 PRINT AT a+i,b+1: PAPER 5:
INK 1: BRIGHT 1:CHR$(79+64*(f$(
i,1)=1)):CHR$(79+64*(f$(i,2)=
1)):CHR$(79+64*(f$(i,3)=1)):
CHR$(79+64*(f$(i,4)=1)):CHR$(
79+64*(f$(i,5)=1)):CHR$(79+
64*(f$(i,6)=1)):CHR$(79+64*(f
$(i,7)=1)):CHR$(79+64*(f$(i,8
)=1)):
4062 LET i=i+1
4064 NEXT j
4066 LET c$=INKEY$
4068 IF c$="C" THEN GO TO 4000
4070 IF c$="0" THEN BEEP 0.02,30
4072 BEEP 0.03,40: LET s$="0000": G
O SUB 3118: GO TO 4002
4074 IF c$="5" THEN BEEP 0.01,xz
+1:yz+1: IF xz>1 THEN LET xz=xz-
1: LET s=1: LET no=1: GO TO 4310

```

```

4250 IF c$="6" THEN BEEP 0.01,xz
+1:yz+1: IF yz<8 THEN LET yz=yz+
1: LET s=2: LET no=1: GO TO 4310
4260 IF c$="7" THEN BEEP 0.01,xz
+1:yz+1: IF yz<1 THEN LET yz=yz-
1: LET s=3: LET no=1: GO TO 4310
4270 IF c$="8" THEN BEEP 0.01,xz
+1:yz+1: IF xz<8 THEN LET xz=xz+
1: LET s=4: LET no=1: GO TO 4310
4280 IF c$="9" THEN BEEP 0.04,30
GO SUB 4300
4290 IF c$="9" THEN GO TO 4600
4300 GO TO 4350
4310 IF s=1 THEN PRINT OVER 1: P
APER 5: INK 1: BRIGHT 1:AT a+yz,
b+xz+1;" "
4320 IF s=2 THEN PRINT OVER 1: P
APER 5: INK 1: BRIGHT 1:AT a+yz-
1,b+xz;" "
4330 IF s=3 THEN PRINT OVER 1: P
APER 5: INK 1: BRIGHT 1:AT a+yz+
1,b+xz;" "
4340 IF s=4 THEN PRINT OVER 1: P
APER 5: INK 1: BRIGHT 1:AT a+yz,
b+xz-1;" "
4350 PRINT OVER 1:AT a+yz,b+xz;
FLASH 1: BRIGHT 1;" "
4370 GO TO 4230
4380 LET d$=SCREEN$(a+yz,b+xz)
4390 IF d$="0" THEN PRINT AT a+y
z,b+xz;"0": LET f$(yz,xz)="0": R
ETURN
4400 IF d$=" " THEN PRINT AT a+y
z,b+xz;"0": LET f$(yz,xz)="0": R
ETURN
4802 GO SUB 3460: LET i=1
4804 FOR j=0 TO 1792 STEP 256
4812 POKE Ram+abc+cba+x+(y+por+r
op-yy-yy)*32+j,VRL ("BIN"+f$(i
))
4816 BEEP 0.02,4+i: LET i=i+1
4818 NEXT j
4820 RANDOMIZE USR 3438: PRINT #
1:AT 0,1;"Poziční kurzoru x=";
X;" y=";Y+por
4824 PRINT #1:AT 1,1;"Chceš pokr
ačovat v zápisu A/N?"
4826 IF INKEY$="N" THEN BEEP 0.0
6: BEEP 0.09,10: GO SUB 4860:
GO TO 2820
4828 IF INKEY$="A" THEN GO SUB 4
840: GO TO 4834
4830 GO TO 4826
4832 LET x=x+1
4834 FOR w=8 TO 0 STEP -1: BEEP
0.01,RND*10-w: NEXT w
4836 GO SUB 4860
4838 PRINT BRIGHT 1: FLASH 1: OV
ER 1: PAPER 8: INK 8:AT y+sk+por
,x;" "
4840 PRINT OVER 1: INK 8: PAP
ER 8:AT y+sk+por-1 AND x=0)-(21
AND x=31 AND y=21),x-1-(32 AND
x=0)-(21 AND x=31 AND y=21);" "
GO TO 4002
4842 IF x=31 AND por=2 THEN LET
x=0: LET y=0: LET rop=0: LET p=
0: LET ab=0: LET yy=0: GO SUB
3312: GO SUB 3232: RETURN
4844 IF x=31 AND y+rop>21 THEN
LET x=0: GO SUB 3284: RETURN
4846 IF x=31 AND y+rop<21 THEN L
ET x=0: LET y=y+1: RETURN
4850 GO TO 4832
4860 GO SUB 3312: POKE 23624,PEE
K (Ram+6200)
4864 IF por=1 OR por=2 THEN RAND
OMIZE USR 3190: RANDOMIZE USR 31
90
4866 RETURN
5000 REM HELP
5002 CLS
5010 PRINT "BARVA: - změna bare
vných param-
etrovce."
5020 PRINT "COPY - tisk na ZX
PRINTRU."
5030 PRINT "DEL: - s CAPS SHI
FT maže dis-
SHIFT maže
atributy."
5040 PRINT "EXCH: - změna obra
zovky."
5050 PRINT "INPUT: - vstup údaj
ů obrazu
ly SLS-vrat
vymaz
UIT-zpět
FT Císelné a
znakové ho
dnoly
5060 PRINT #1:AT 0,7;"Stiskněte
klávesu"
5065 PAUSE 0
5070 CLS
5080 PRINT "BEU: - s CAPS SHI
FT maže komp-
zovky."
5090 PRINT "DEM: - s CAPS SHI
FT - záznam
ozi kurzoru
eti.
SHIFT zápis
o obrazovky."
5100 PRINT "OVER: - měnění ob
razovce."
5110 PRINT "QUIT: - návrat do
hlavní
5120 PRINT "ROLL: - posun obra
zu do směru
h kláves
FT kloust
SHIFT tri-
buty."
5130 PRINT #1:AT 0,7;"Stiskněte
klávesu"
5140 PAUSE 0: GO TO 2
5000 CLEAR 39999: LOAD "CODE"
RUN
7000 REM UDG
7001 REM ASCIIČÍZNAČYUSU0A0R0F0U
ABCDEF0GHIJ0KLMNOP0RSTU

```

```

1 REM SKICAK s1r0j.kod + UDG
10 FOR i=64599 TO 65534
20 READ a: POKE i,a
30 NEXT i

110 DATA 243,006,192,033,000,00
4,205,178,014,201,033,255,007,01
7,223,087,229,213,014,023,006,03
2,026,119,121,230,007,254,001,03
2,002,151,018,043,027,016,241,01
3,040,021,121,230,007,254,000,04
0,024,254,007,032,225,213,017,00
0,007,167,237,082,209,024,215,20
9,225,021,037,124,254,079,200,02
4,201,229,033,000,087,235,107,23
7,082,235,024,193,033,223,09
0,017,255,000,001,224,002,237,16
4,052,056,000,032,018,027
120 DATA 016,252,203,017,032,00
0,004,025,016,253,055,035,016,25
3,209,056,197,006,000,075,213,22
9,017,032,000,167,237,082,209,22
9,237,184,225,209,193,120,016,23
4,241,067,119,043,016,252,201,01
7,064,183,033,000,054,001,000,02
7,237,176,201,017,064,183,033,00
0,064,006,027,197,006,000,000,00
0,000,026,000,000,000,119,035,01
9,000,016,243,193,016,237,201,01
7,028,028,033,000,054,006,027,19
7,006,000,126,245,026,119
130 DATA 241,018,035,019,016,24
6,193,016,240,201,033,000,064,00
0,024,197,006,000,126,238,255,11
9,035,016,249,193,016,243,201,01
9,066,033,015,003,017,026,001,06
9,233,197,067,062,022,215,124,21
5,122,215,121,033,000,016,244,03
6,193,009,016,236,000,082,000,23
0,007,087,033,000,088,005,003,19
7,006,000,126,230,248,178,119,03
5,016,248,193,016,242,201,062,02
2,203,033,203,039,203,039,230,05
6,087,033,000,088,006,003
140 DATA 197,006,000,126,230,19
9,178,119,035,016,246,193,016,24
2,201,000,001,000,024,017,064,15
6,033,000,064,026,182,119,035,01
9,011,120,177,032,246,201,033,00
0,064,085,062,192,006,031,035,09
4,043,115,035,018,249,114,035,06
1,032,242,201,033,255,087,022,00
0,062,192,006,031,043,094,035,11
0,043,016,249,114,043,061,032,24
2,201,033,025,087,014,192,006,03
6,184,203,022,043,016,251,013,03
2,245,000,033,000,064,014
150 DATA 192,006,033,013,203,03
0,035,016,251,013,032,145,201,03
3,000,064,017,000,065,014,192,00
6,032,026,119,121,254,002,032,00
2,151,018,019,035,016,243,213,01
7,224,000,025,227,025,235,225,01
3,121,230,007,254,000,032,018,21
3,017,224,007,167,237,082,209,02
4,014,254,001,032,010,229,235,01
7,224,007,167,237,082,235,225,12
1,230,063,254,000,032,006,062,00
7,132,103,024,187,254,001,032,18
3,062,007,130,087,121,254
160 DATA 001,032,174,201,033,00
0,088,042,056,014,024,006,031,03
3,094,043,115,035,016,249,119,03
0,013,032,242,201,033,255,090,06
2,056,014,024,006,031,043,094,03
1,115,043,016,249,119,043,013,03
2,242,201,033,032,088,017,000,08
0,001,224,002,237,176,062,056,08
0,019,019,019,255,086,014,192,00
3,255,087,017,255,086,014,192,00
0,032,026,119,121,254,002,032,00
2,151,018,027,043,016,243,213,01
7,224,000,167,237,082,227
170 DATA 167,237,082,235,225,01
3,121,230,007,254,000,032,008,21
3,017,224,007,025,209,024,011,25
4,001,032,007,229,033,224,007,02
5,235,225,121,230,063,254,000,03
2,006,124,214,007,183,024,188,25
4,001,032,184,122,214,007,087,12
1,254,001,032,175,201,033,000,06
4,017,032,064,229,213,014,023,00
6,032,026,119,121,230,007,254,00
1,032,002,151,018,035,019,016,24
1,013,040,019,121,230,007,254,00
0,040,022,254,007,032,225
180 DATA 213,017,000,087,025,08
9,024,217,209,225,020,036,024,25
4,072,032,204,201,229,033,000,08
7,025,235,225,024,193,033,000,08
8,001,000,003,062,056,119,035,01
1,120,177,032,247,201,033,000,06
4,001,000,024,062,000,119,035,01
1,120,177,032,247,201,198,030,06
3,156,118,027,003,019,000,062,00
0,016,056,004,060,068,050,000,04
0,016,124,008,016,032,124,000,02
0,008,028,032,032,032,028,000,00
0,005,006,006,068,068,068
190 DATA 000,008,016,056,068,12
0,064,060,000,060,060,066,064,05
4,066,060,000,060,126,004,008,01
6,032,126,000,008,016,068,068,05
0,060,004,056,008,016,000,048,01
6,016,056,000,008,016,068,068,05
0,060,056,000,060,060,064,060,08
2,066,060,000,008,082,066,066,08
0,060,060,000,060,120,066,066,08
0,068,068,000,008,016,120,068,06
0,068,068,000,008,016,066,068,06
4,068,068,000,060,124,066,066,12
4,068,068,000,060,016,056
200 DATA 068,120,064,060,000,02
0,008,028,032,032,032,032,000,04
0,016,056,064,056,004,120,000,01
0,020,016,056,016,016,120,000,01
6,000,068,068,068,068,056
1000 SAVE "code"CODE 64599,936
1010 VERIFY "code"CODE 64599,936

```

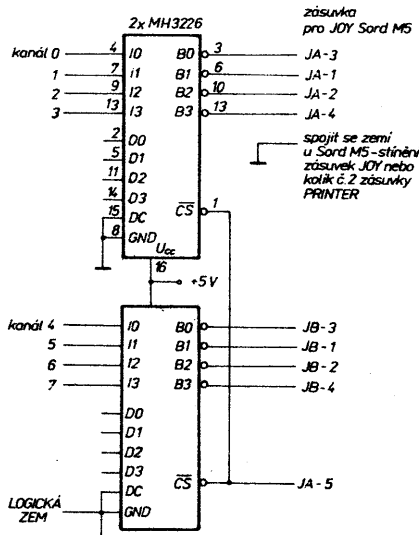
ANALOG

Miloš Skopec, U Santošky 11, 150 00 Praha 5

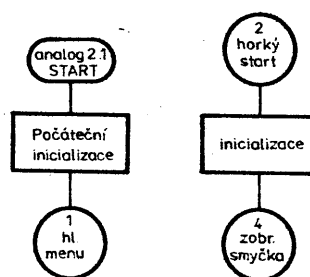
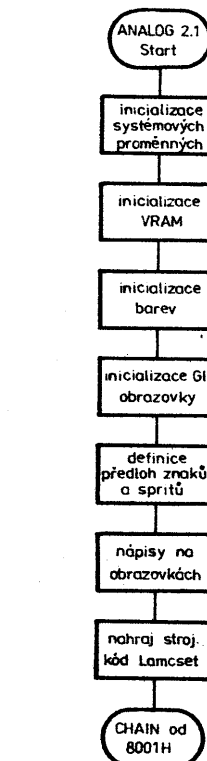
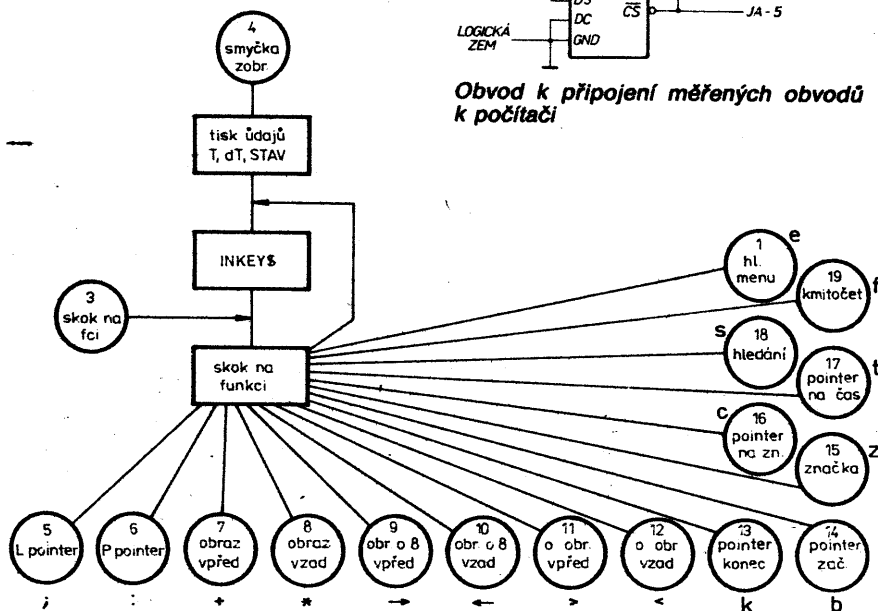
Program ANALOG 2.1 umožňuje použít počítač Sord M5 jako logický časový analyzátor (k počítači musí být připojen modul Basic F a EM-5). Umožňuje záznam a zobrazení časových průběhů osmi nezávislých logických signálů metodou vzorkování. Tato metoda a její nevýhody je podrobně popsána v [1]. Při záznamu čte počítač v určitých časových intervalech (doba vzorkování, Tv) data ze vstupního portu a ukládá je do paměti (24 kB). Po naplnění paměti lze zobrazit časové průběhy signálů na obrazovce ve tvaru časového diagramu.

Při záznamu je možno volit vstupní port, vzorkovací dobu, podmínku začátku nebo ukončení záznamu a typ záznamu (se spuštěním nebo s ukončením). Při zobrazování se lze „procházet“ celou záznamovou pamětí, měřit délku časových úseků a kmitočety jednotlivých signálů, automaticky vyhledávat různé kombinace úrovní atd. Zaznamenaná data se dají uchovávat na magnetofonové kazetě. Během zobrazování je uživatel průběžně informován o okamžitém čase a logickém stavu.

Program se skládá ze tří dílčích programů, které se postupně nahrávají do počítače. První program inicializuje systém, definuje předlohy znaků, obsazení VRAM atd. Dále nahrává druhý program ve strojovém kódu (1 kB),



Obvod k připojení měřených obvodů k počítači



který je uložen od adresy 7900H a obsahuje pomocné rutiny pro záznam a zobrazení. Nakonec nahrává vlastní obslužný program, který je opět v BASICu. Ten zajišťuje ve spojení s podprogramy ve strojovém kódu všechny funkce logického časového analyzátoru.

Parametry analyzátoru:

Doba vzorkování: se spuštěním 6,14 μ s, 20 μ s až 566,7 μ s (krok po 2,22 μ s), s ukončením —31,1 až —566,7 μ s (krok po 2,22 μ s).

Počet kanálů: 8.

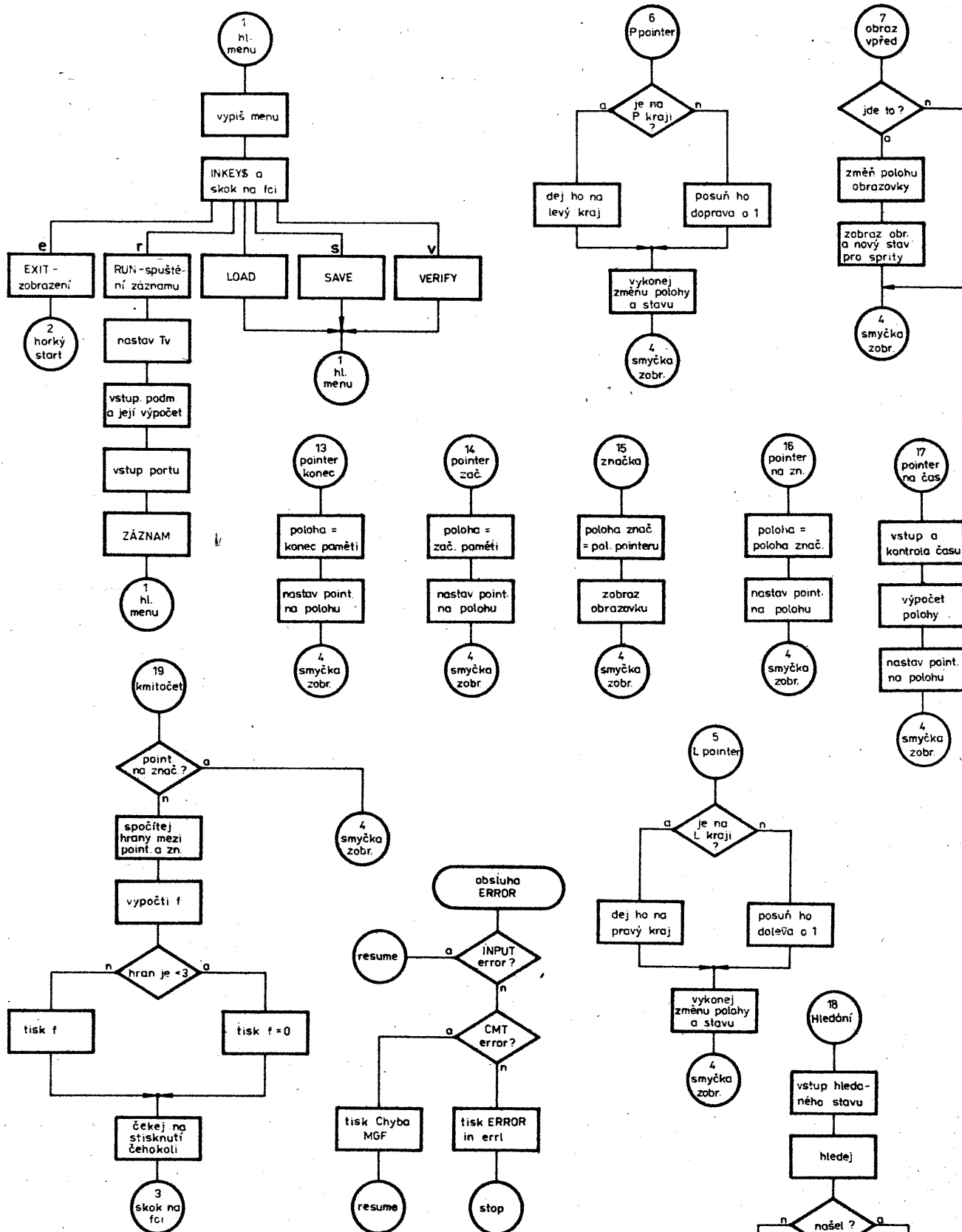
Rozsah záznamové paměti: 24 kB.

Podmínka spuštění nebo ukončení: lze zadat u jednotlivých kanálů log. 0, log. 1, nebo bez významu.

Po nahrání všech tří částí programu se program ohlásí nápisem **READY**. Nyní můžeme buď spustit záznam, zobrazit časový diagram nebo pracovat s magnetofonem.

Při funkci záznam nastavíme nejprve vzorkovací dobu pomocí kurzorových šipek. Nastavená doba se průběžně zobrazuje vlevo nahoře. Po nastavení požadované doby stiskneme RETURN. Poté určíme, zda chceme podmínku (a/n) a (přip.) tuto podmínku v binárním tvaru. Nakonec vložíme adresu vstupního portu (pozor — dekadicky, nebo před hex. číslo napsat znak &). Prakticky připadá v úvahu port &50 (analýza signálu z magnetofonu) a port &37 (pro použití měřicího interfejsu — viz obr.). Při Tv=6,14 μ s lze pro testování nebo demonstraci použít port 1 (čítač CTC #1). Nyní, pokud máme nastaveno

spuštění podmínkou, se spustí záznam po splnění této podmínky nebo po stisknutí tlačítka RESET. Paměť se zaplní vstupními daty a vypíše se opět READY. Nyní máme opět možnost si vybrat z hlavního menu. Pokud bylo nastaveno ukončeno podmínkou (Tv<0), pak se po zadání portu nejprve zaplní celá paměť daty (na vyšších rozsazích to trvá až několik sekund) a pak se teprve testuje stisknutí RESET nebo splnění ukončovací podmínky. Během toho se paměť stále cyklicky plní daty. Po RESET nebo splnění podmínky se záznam zastaví a opět se vypíše READY.



Při funkci zobrazení se na obrazovce objeví časové průběhy signálů. Uprostřed obrazovky je svislá přerušovaná čára — ukazatel. Při levém kraji obrazovky se nachází podobná čára — značka. Ukazatelem můžeme pohybovat tlačítky ; a : . Značku můžeme umístit na žádanou pozici tak, že na

tuto pozici umístíme ukazatel a stiskneme tlačítko z. Tím umístíme značku na toto místo. Tlačítky + a * se můžeme s obrazovkou „pohybovat“ po paměti, tlačítka ← a → se používají pro rychlejší pohyb po osmi bodech a konečně tlačítka < a > umožní pohyb vpřed nebo vzad o celý rozsah obrazovky (256 bodů). Vlevo od ukazatele jsou zelená čísla, označující číslo signálu. Vpravo je číslo, které určuje okamžitou logickou úroveň v místě ukazatele — červená jednička nebo

modrá nula. Na křivkách jednotlivých kanálů je log. 1 vybarvena červeně, log. 0 modře a hrany žlutě.

Nahoře na celním panelu se průběžně zobrazuje okamžitý stav v místě ukazatele v hexadecimálním tvaru. Pod ním je zobrazen čas v milisekundách v místě ukazatele a délka intervalu mezi ukazatelem a značkou. Pomocí tohoto intervalu lze snadno měřit délku impulsů nebo posun hran jednotlivých signálů.

Dále umožňuje analyzátor v režimu zobrazení tyto funkce:

t — po zadání času v milisekundách se přenesou ukazatel do příslušného místa,
z — umístí značku na místo, kde je ukazatel,
c — přenesou ukazatel na místo, kde je značka,
b — umístí ukazatel na začátek paměti,
k — umístí ukazatel na konec paměti,
s — po zadání stavu (dekadicky nebo &hex) přesune ukazatel na první místo, kde je tento stav. Prohledává se od místa kurzoru do konce paměti,
f — po zadání čísla kanálu vypočítá průměrný kmitočet signálu v intervalu mezi ukazatelem a značkou (včetně jejich poloh). Kmitočet se vypočítá z počtu hran v tomto intervalu, proto je vhodné pro větší přesnost nastavit jak ukazatel, tak značku na hranu u toho kanálu, kde chceme měřit kmitočet.

e — ukončí režim zobrazení a umožní opět výběr v hlavním menu.

Poznámky k použití programu:

Protože program mění obsah některých systémových proměnných a uspořádání VRAM, je vhodné po skončení práce s ním inicializovat systém (CALL 0, vypnutí počítače). Pokud chceme během programu přejít do BASICu (např. změna rychlosti nahrávání na MGF), lze tak beztržně učinit v režimu hlavního menu po vypsání READY stisknutím CTRL+RESET. Pokračovat v programu lze příkazem CONT. Upozorňuji, že druhý program v BASICu není v paměti umístěn jako obvykle od adresy 77F2H, ale kvůli strojovému programu se nahrává od adresy 8001H (zajišťuje inicializační basicový program).

Závěr

Popsaný program prokáže velmi dobré služby při analýze a optimálním nastavení signálu z magnetofonu a při oživování různých periférií a jiných číslicových zařízení, kde se nepoužívají kmitočty vyšší, než asi 100 kHz (např. dálkopis, A/D převodník s C 520D ...).

Literatura

[1] Úvod do techniky logických analyzátorů. Ročenka sdělovací techniky — Praha 1987.

Seznam proměnných hlavního programu a jejich význam:

celočíslné:

CTC ... časovací konstanta pro časovač při záznamu (8 až 255),
SIGN ... 1..startovací podmínka, 0..ukončovací podmínka,
G ... počet míst za desetinnou čárkou pro tisk času (2 až 4),
HL ... poloha obrazovky v paměti &A000 — &FFFF,
LAB ... poloha značky v paměti &A000 — &FFFF,
X ... x-ová souřadnice ukazatele na obrazovce (0 až 255),
T ... pomocná proměnná,
Q ... podmínky startu nebo konce záznamu (0 až 255),
E ... maska podmínky startu nebo konce záznamu (0 až 255),
I ... parametr cyklů,
C ... číslo vstupního portu (0 až 255).

reálné:

BASE# ... vzorkovací doba v milisekundách — 566.7E-3 až +566.7E-3,
T# ... pro vstup času.

řetězcové:

A\$... pro vstup povelů,
B\$... pro vstup podmínky v binárním tvaru.

```
1000 LOGICKY CASOVY ANALYZATOR
1010 -----
1020
1030 SCOPWARE COPYRIGHT 1987
1040
1050 VERZE 2.1

1060
1070
1080 Inicializace systemu
1090 -----
1100
1110 type int:clear 256,&9FFC:poke &701A
,&94:poke &7033,3:poke &9FFD,1,9
1120
1130 Inicializace VRAM
1140 -----
1150
1160 print "UZSIVTR:call &0E01,0,&1000,
,&0000:call &0E01,&1E00,&0800,&0call &0
&81,,&2000,&1800:call &0E01,0,&0800,,&2
&00:call &0800,,&0001,,&9FFF:call &0E01,
&9FF0,&0100,,&3A00
1170 poke &70BD,&2000:poke &709D,&2000
1180
1190 Inicializace barev
1200 -----
1210
1220 for I=&0C00 to &13FF step 8
1230 vpoke I,&81,&B1,&B1,&B1,&B1,&B1,&B1
,&71
1240 next I
1250
1260 Inicializace GII obrazovky
1270 -----
1280
1290 K=&80
1300 for I=&3900 to &3AFF step &40
1310 for J=I to I+&1F
1320 vpoke J,(K and &FF)
1330 K=K+1
1340 next J,I
1350
1360 Definice znaku a spritu
1370 -----
1380
1390 view 0,0,31,7:cls:X=128
1400 for I=&A2 to &C5:stchr "0000000000
000000" to I:next
1410 stchr "8000000000000000" to &A0:stc
hr "8000000000000000" to &A1:stchr "2
1420 for I=&16 to 31:scod I,&A0:scod I,&15
+(&123):loc I to X,&64+16*(I mod 8):next
1430 stchr "0007050505070000" to &A5
1440 stchr "0002020202020000" to &A7
1450 stchr "0020505050200000" to &A0
1460 stchr "0020602020200000" to &B1
1470 stchr "0060902040f00000" to &B5
1480 stchr "0002000200000000" to &B9
1490 stchr "0020600f02000000" to &BD
1500 stchr "0008000200000000" to &C1
```

```
1510 stchr "0008000000000000" to &C5
1520 stchr "0002040404000000" to &C9
1530 for I=0 to 7:scod I,&4+I:&AC:scod I,
2:loc I to X,&64+16*I:next
1540 for I=8 to 15:scod I,&A4:scod I,&4:1
oc I to X,&64+16*(I-8):next
1550 stchr "002040f040200000" to &B2,1
1560 stchr "000804f040800000" to &B3,1
1570 stchr "0000004848784848" to &B1,1
1580 stchr "3c4299a1a199423c" to &B0,0
1590 stchr "0000000000010000" to &B4,0
1600 stchr "0000000000281000" to &B5,0
1610
1620 Tisk obrazovek
1630 -----
1640
1650 print "H STAV : 0001 SCOPWARE 1987
H T= 0.000 ms:dt= 0.000 msW:rp:tr(3
2,-):Menu: (<=>:e) t,b,k,s,c,z,f,e
" :cursor(0,7):rp:tr(32,-):"KC"
1660 print "I:scod I,&0CA3
1670 print "I v d vI vM
LOGICKY CASOVY ANALYZATORW:rp:tr(38,"
-):"I T= 0.000 sI P=-----II=000
IIII 76543210111111:rp:tr(38,-):"I
" :rp:tr(38,-):"I
1680 print "I I I A v I I I r...SP
USTENII I I I I I I I I s,v...
LOAD,SAVE,VERIFY I I I I I I
I I e...ZOBRAZENII I I I I I
I I :rp:tr(38,-):"I
1690 view 0,16,39,23
1700
1710 Basic od 8000H, nahraj cm a CHAIN
1720 -----
1730
1740 poke &8001,&FF:poke &726A,&8002,&8
004,&8004:old "LAMset?":chain

1000 LOGICKY CASOVY ANALYZATOR
1010 -----
1020
1030
1040 Pocatecni inicializace
1050
1060 CTC=9:BASE#=0.02#:SIGN=1:G=4
1070 cision error gosub HERR:goto REXIT
1080
1090 Horky start zobrazovani
1100
1110 RHOT
1120 HL,LAB=&A000:X=128:goto RZOBH
1130
1140 Smycka zobrazovani
1150
1160 RLOOP
1170 print cursor(8,1):right$(hex$(peek(
HL+X)),2):cursor(3,3):num$(HL+X+&5FFF*5
IGN+1)*abs(BASE#),8-G):cursor(19,3):nu
m$(HL+X-LAB)*BASE#,8-G,G)
1180 RKEY:AR=key$R
```

```
1190 RCOMMAND
1200 if AR="" then goto RKEY else if AR=
"i" then goto RSPD else if AR="s" then g
oto RSTI else if AR="t" then goto RST c
lse if AR="e" then goto RSTT else if AR=
"e" then goto RFSB else if AR="4" then g
oto RFSB
1210 if AR="<" then goto RPSD else if AR
=">" then goto RPSI else if AR="t" then
goto RTIME else if AR="s" then goto RSEA
R else if AR="e" then goto REXIT else if
AR="c" then goto RCAR else if AR="z" th
en goto RZNAC
1220 if AR="f" then goto RFRQ else if AR
="k" then goto RKONEC else if AR="b" th
en goto RBEGIN else goto RKEY
1230
1240 Posunuti pointeru vlevo
1250
1260 RSPD:if X<1 then X=255 else X=X-1
1270 call &7AE0,,,X:call &7AFC,,,peek(H
L+X):goto RLOOP
1280
1290 Posunuti pointeru vpravo
1300
1310 RPSI:if X>254 then X=0 else X=X+1
1320 call &7AE0,,,X:call &7AFC,,,peek(H
L+X):goto RLOOP
1330
1340 Posunuti obrázku vřed
1350
1360 RSTT:if HL>257 then goto RLOOP else
HL=HL+1:call &7A9E,,,LAB-HL,HL:call &7A
FC,,,peek(HL+X):goto RLOOP
1370
1380 Posunuti obrázku vzad
1390
1400 RSTI:if HL<24575 then goto RLOOP el
se HL=HL-1:call &7A9E,,,LAB-HL,HL:call &
7AFC,,,peek(HL+X):goto RLOOP
1410
1420 Posunuti obrázku vřed o osm bodu
1430
1440 RFSB:if HL>264 then goto RLOOP else
HL=HL+8:call &7B81,,,LAB-HL,HL:call &7A
FC,,,peek(HL+X):goto RLOOP
1450
1460 Posunuti obrázku vzad o osm bodu
1470
1480 RFSB:if HL<24568 then goto RLOOP el
se HL=HL-8:call &7B81,,,LAB-HL,HL:call &
7AFC,,,peek(HL+X):goto RLOOP
1490
1500 Posun o obrázku vřed
1510
```

```

2250 if AR=":" then if SIGN=0 and CTC<0 and CTC>=1 then CTC=CTC-1 else if SIGN=0 then SIGN=1:CTC=8 else if CTC<255 then CTC=CTC+1
2260 if CTC=8 then BASE#=0.00614# else BASE#=CTC*2.222222222222E-3#
2270 if SIGN=0 then BASE#=-BASE#
2280 goto RDISPVT
2290RHOTOTOV
2300 poke &9FFD,SIGN,CTC;if CTC<19 then G=4 else if CTC<184 then G=3 else G=2
2310'
2320' Vstup a vypocet podminky
2330'
2340RPODM:Q,E=0:view 0,16,39,23:input "H vWPODMINKA : (a,n),[&x(0,1,x)]":AR,B
R
2350 if AR="#" then BR=rprt&(0,"-"):Q=&FF:goto RDISPOD
2360 if len(BR)>&S then goto RPODM
2370 for I=0 to 7
2380 AR=mid$B,BR,8-I,1)
2390 if AR="0" then E=E+2*I:goto RDALSI
2400 if AR="1" then E=E+2*I:Q=Q+2*I:goto RDALSI
2410 if AR<>"x" then I=10
2420RDALSI:next I
2430 if I=10 then goto RPODM
2440RDISPOD:view:print cursor(20,3);BR:v
iew 0,16,39,23
2450'
2460' Vstup vstupniho portu
2470'
2480RINPORT:input "KWXEXPORT ":C
2490 if C<0 or C>255 then goto RINPORT
2500 view:print cursor(34,3):right$(hex$(C,2)):view 0,16,39,23
2510'
2520' Vlastni spusteni sberu dat
2530'
2540 print "HA v ^ ^ ^ wWCekani na
spusteni":if AR="n" then print "RESET."
else print "podminka nvrnou."
2550 if SIGN=0 then call &78FF,C,Q*256+C,Q*256+E else if CTC>8 then call &7900,C,Q*256+C,Q*256+E else if AR="n" then c
all &7ABF,C else call &7962,C,Q*256+E
2560 poke &9FFF,peek(&A000):goto REXIT
2570'
2580' Mereni kmitoctu
2590'
2600HFRO
2610 if HL=X-LAB then goto HLOOP else vi
ew 0,6,31,6:input "Linka (0..7) = ":I:if
T<0 or T>7 then goto HFRO else if HL+X
LAB then call &78DD,T*256,,HL+X-LAB+1,LA
B else call &78DD,T*256,,HL-HL-X+1,HL+X
2620 if reg(3)<2 then pokew &7268,0
2630 print "f";Ts ="numr(reg(3)*500#/a
bs(BASE#)/abs(LAB-HL-X),7,2)":Hz"
2640'
2650' Cekani na stisknuti klavesy
2660'
2670RCEKEJ:repeat:AR=inkey$:until AR<>" "
2680 print:view 0,0,31,7:goto RCOMMAND
2690'
2700' Obsluzeni chyby INPUT nebo CNT
2710'
2720ERR:if err=25 then resume else if e
rr=18 then print "NGCHYBA MGF !":sleep 1
resume
2730 print "Err";err;"in";errl

```

[illegible]

RYCHLY SBER DAT BEZ PODMINKY

```

; VSTUP : C...port
;
;
;
FSBREG: DI          ;zказ INT
LD      HL,0A00H ;odkud RAM
LD      B,0      ;256x
FSBRE1: IN          A,(50H)
RLCA    ;bylo RESET?
JP      NC,FSBRE1 ;nebylo
JP      FSBRE1    ;sbirej

```

ZOBRAZENI JEDNE OBRAZOVKY

SCOPWARE /C/ 1987

VSTUP: DE...adresa znacky
HL...odkud zobrazit

```

;
;
;
OBRZ: DI          ;zказ INT
PUSH    DE        ;adr.znacky
PUSH    HL        ;odkud->IX
POP      IX
LD      E,0       ;od 0. sloupce
OBR1: CALL  OBRSLD ;sloupec
INC      E        ;dalsi
BIT      5,E      ;vsech 32?
JR      Z,OBR1    ;jeste ne
POP      HL        ;adr.znacky
CALL    ZOBZNA    ;zobraz znacku
RET

```

ZOBRAZENI ZNACKY

VSTUP: HL...poloha znacky

```

;
;
;
ZOBZNA: LD      A,H ;hl>255?
OR      A
JR      Z,DRAW3 ;neni
LD      HL,-1    ;nekresli
DRAW3: LD      A,51 ;od spritu
LD      B,8      ;8 spritu
PMOVE: PUSH    BC
PUSH    HL
PUSH    AF
CALL    042BH ;GTSP0S
POP      AF
POP      HL
PUSH    AF
CALL    03CEH ;MVSPA
POP      AF
POP      BC
DEC      A
DJNZ    PMOVE   ;dalsi sprite
EI      ;povol INT
RET      ;navrat

```

RESULT: DS 16 ;pro vysledky

POSUNUTI SPRITU PO OSE X

VSTUP: HL...nova x-ova sour.

```

;
;
;
SPRSUN: LD      A,23 ;poc.spr.
MOVE: PUSH    HL
PUSH    AF
CALL    042BH ;GTSP0S
POP      AF
POP      HL
PUSH    AF
CALL    03CEH ;MVSPA
POP      AF
CP      8 ;je 8.sprite?
JR      NZ,MOV1 ;neni
LD      DE,0008H ;rozdil
SBC     HL,DE ;poloh x
MOV1: SUB     1 ;dalsi sprite
JR      NC,MOVE ;jeste opakuji
RET      ;uz ne

```

PREVOD NA BIN. TVAR PRO SPRITY

VSTUP: E...cislo

```

;
;
;
PREVOD: LD      B,8 ;8 bitu=byte
PREV: LD      D,B ;schovej citac
LD      HL,PREDL ;predlohy
RLC      E
JR      NC,PREV1 ;bit=0
INC      HL ;druha predl.
INC      HL
PREV1: LD      C,(HL) ;vezmi predl.
INC      HL
LD      B,(HL) ;a uloz ji

```

```

LD      A,D ;cislo spritu
ADD     A,7 ;vypocti
CALL    0454H ;STSCOD
LD      A,D ;c.spritu
ADD     A,7 ;vypocti
CALL    0445H ;STSCOL
LD      B,D ;parametr
DJNZ    PREV ;8x
RET      ;konec

```

PREDL: DS 0A4H,05H,0A5H,08H

HLEDANI a OD HL, DELKA=bc

```

;
;
;
HLEDEJ: DI          ;zказ INT
CPIR    ;hledej
EI      ;povol INT
RET      ;vrat se

```

ZOBRAZENI JEDNOHO SLOUPCE (8bit)

VSTUP: E...cislo sloupce
IX...odkud v pameti

```

;
;
;
OBRSLD: LD      HL,390BH ;L.H.roh
PUSH    DE ;schovej E
CALL    14C5H ;GBVRAM
ADD     A,E ;vypocet prvnj
AND     1FH ;adresy VRAM
RLCA    ;E. sloupce
RLCA

```

```

LD      E,A ;lsb->E
LD      A,(IX-1) ;minuly stav
LD      B,8 ;8 byte
TRANS1: LD      C,B ;schovej citac
LD      HL,RESULT ;vysledek
LD      D,(IX) ;data
XOR     D ;hrany
LD      B,8 ;8 bitu
TRANS2: RRC     D ;vezmi bit
RL      (HL) ;zaroluj
INC      HL
RRCA    ;zaroluj hrany
RL      (HL) ;do vysledku
INC      HL ;dalsi vysled.
DJNZ    TRANS2 ;cely byte
INC      IX ;dalsi data
LD      A,D ;minula data
LD      B,C ;citac bytu
DJNZ    TRANS1 ;dalsi byte
LD      D,6CH ;msb adr. VRAM
LD      BC,0811H ;port 11 a 8x
LD      IV,RESULT ;vysledek
LD      L,B ;schovej citac
OUT     (C),E ;adresu do VDP
OUT     (C),D
LD      A,(IV) ;data
LD      B,(IV+1) ;hrany

```

TRANS1: LD C,B ;schovej citac
LD HL,RESULT ;vysledek
LD D,(IX) ;data
XOR D ;hrany
LD B,8 ;8 bitu
TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

TRANS2: RRC D ;vezmi bit
RL (HL) ;zaroluj
INC HL
RRCA ;zaroluj hrany
RL (HL) ;do vysledku
INC HL ;dalsi vysled.

```

LD      D,A ;bity zpet
EX      AF,AF ;posledni znak
LD      B,C ;citac radek
DJNZ    LSUN81 ;dalsi radek
LD      DE,24H ;posled.sloupec
ADD     IX,DE ;vpravo
LD      E,1FH ;to je on
JR      SKONCI ;dokonci

```

POSUN OBRAZOVKY VPRAVO O 8

VSTUP: jako LSUN8

```

;
;
;
RSUN8: PUSH    DE ;adresa znacky
PUSH    HL ;odkud->IX
POP      IX
CALL    PRIPRA ;inicializace
DEC      A ;novy znak
RSUN81: LD      C,B ;citac radku
LD      B,E ;del.radku=20H
RSUN82: AND     1FH ;modulo 32
OR      D ;nejvys. 3 bity
CALL    14BDH ;PBVRAM
INC      HL ;adr.dals.znaku
INC      A ;dalsi znak
DJNZ    RSUN82
EX      AF,AF ;schov.posl.zn.
LD      A,D ;zvets.nejvys.
LD      D,0 ;3 bitu o 1 a
ADD     HL,DE ;vypocet adresy
ADD     A,E ;pro dal. radek
LD      D,A ;bity zpet
EX      AF,AF ;posl.znak zpet
LD      B,C ;citac radku
DJNZ    RSUN81 ;novy radek
LD      E,0 ;1. sl. zleva
SKONCI: CALL  OBRSLD ;kresli sloupec
POP      HL ;adresa znacky
JP      ZOBZNA ;kresli znacku

```

```

;
;
;
SKONCI: CALL  OBRSLD ;kresli sloupec
POP      HL ;adresa znacky
JP      ZOBZNA ;kresli znacku

```

```

;
;
;
PRIPRA: DI          ;zказ INT
LD      HL,390BH ;adr. 1. znaku
LD      B,8 ;8 radku
CALL    14C5H ;GBVRAM
LD      DE,0820H ;3b a del.rad.
RET      ;vrat se

```

MERENI PRUMERNEHO KMITOCTU

VSTUP: A...linka 0-7
DE...delka useku RAM
HL...odkud v RAM

VYSTUP: HL...pocet hran -1

```

;
;
;
MERPER: DI          ;zказ INT
INC      A ;vypocet masky
LD      B,A
LD      A,0BH
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

;
;
;
MERLP: RLCA
DJNZ    MERLP
PUSH    HL
POP      IX ;odkud->IX
LD      HL,-1 ;pocet hran-1
LD      C,A ;maska do C
MER1: LD      A,(IX-1) ;minula data
XOR     (IX) ;por.se soucas.
INC      IX ;dalsi data
DEC      DE ;cit.del.useku
AND     C ;zamaskuj
JR      Z,NEHRAN ;nebyla hrana
HL      ;byla->zvets
OR      E ;DE=0?
JR      NZ,MER1 ;jeste ne
EI      ;povol
RET      ;navrat

```

```

FILL:  HALT      ;cekej, az se
        JP       NZ, FILL ;pamet zaplni
        LD       HL, UPINT2 ;plno-zmen
        LD       (7004H), HL ;proced. INT
        LD       HL, 0A000H ;pocat. adr.
RPT:    IN       A, (50H) ;smycka cekani
        RLCA     ;RESET?
        JP       NC, RPT  ;nebylo
CHALAS: CALL     KONEC   ;ukonci
        POP      HL      ;adr. zacat. dat
        JR       POSUN   ;posun pamet
;
;
UPINT2: EX       AF, AF' ;schovej AF
        IN       A, (C)  ;data
        LD       (HL), A ;uloz
        AND      E       ;maska
        XOR      D       ;podm. splnena?
        JR       Z, VEN  ;ano->skonci
        EX       AF, AF' ;vrat AF
        INC      HL      ;dalsi adresa
        BIT      7, H    ;HL je uz 0?
        JR       NZ, UPINT2 ;ne
        LD       H, 0A0H ;ano->HL=A000
UPI21:  EI       ;povol
        RETI      ;navr. do smycky
VEN:    CALL     186DH   ;IGNORJ
        POP      DE      ;znice navr. adr.
        EX       AF, AF' ;vrat AF
        JR       CHALAS  ;a vrat se
;
;
; POSUN USEKU RAM 0A000H - 0FFFFH
;
; VSTUP: HL...ADRESA, JEJIZ OBSAH
;       PATRI NA ADRESU 0A000H
;
;=====
POSUN:  PUSH     HL
        LD       HL, 2000H ;schov. predl.
        LD       DE, 3000H
        CALL     0B81H ;BLKMV2
        POP      HL
        DI       ;zakaz INT
        LD       A, 0D0H ;pul RAM
        CP       H       ;sktera?
        JR       C, PUL2 ;2. pulka
        PUSH     HL
        LD       DE, 6000H
        ADD      HL, DE  ; -0A000H
;
;
        LD       B, H
        LD       C, L   ;BC=delka
        LD       A, H
        OR       L
        JR       Z, HOTOV ;nesoupej
        LD       HL, 0A000H
        LD       D, L
        LD       E, L   ;DE=0
        CALL     0E61H ;RAM-VRAM
        DI
        LD       H, B
        LD       L, B   ;HL=0
        POP      DE
        PUSH     DE
        SBC      HL, DE
        LD       B, H
        LD       C, L   ;BC=delka
        EX       DE, HL ;HL=odkud
        LD       DIR   ;BC, 6000H
        LD       HL
        POP      HL
        ADD      HL, BC  ; -0A000H
        LD       B, H
        LD       C, L   ;BC=delka
        LD       L, H   ;HL=0
        CALL     0E7DH ;VRAM-RAM
        CALL     JR     OBRINI
        POP      HL
        JR       OBRINI
        XOR      A       ;CY=0
        PUSH     HL
        EX       DE, HL
        LD       H, A
        LD       L, A
        SBC      HL, DE
        LD       B, H
        LD       C, L   ;BC=delka
        EX       DE, HL ;HL=odkud
        LD       D, A
        LD       E, A   ;DE=0=kam
        CALL     0E61H ;RAM-VRAM
        DI
        POP      DE
        LD       HL, 6000H
        ADD      HL, DE
        LD       B, H
        LD       C, L   ;BC=delka
        EX       DE, HL ;HL=odkud
        DEC      HL
        LD       DE, 0FFFFH
;
;
LDDR:   EX       DE, HL
        XOR      A       ;CY=0
        SBC      HL, DE
        LD       B, H
        LD       C, L   ;BC=delka
        INC      DE      ;DE=A000H
        LD       L, A
        LD       H, A   ;HL=0=od
        CALL     0E7DH ;VRAM-RAM
;
;
; INICIALIZACE VRAM PO POSUNUTI
;
;=====
OBRINI: XOR      A       ;A=0
        LD       BC, 2800H ;nulu VRAM
        LD       HL, 0800H ;od barev 2.
        CALL     0E01H ;PADVRM
        LD       A, 1EH ;scer+seda
        LD       BC, 0800H
        LD       HL, 0   ;bar. pro 1. tre.
        CALL     0E01H ;PADVRM
        LD       DE, 2000H
        LD       HL, 3000H ;vrat predlohy
        CALL     0B81H ;BLKMV2
        DI       ;zakaz INT
        LD       HL, 0C00H
        CALL     149FH ;STVWAD
        LD       B, 0    ;barvy pro
        LD       C, 3    ;2. a 3. tretinu
        LD       B, 8
        LD       HL, BARVY
        LD       A, (HL) ;(10H), A
        OUT      (10H), A
        INC      HL
        DJNZ     OBRIN2
        LD       B, C
        DJNZ     OBRIN1
        EI       ;uz povol
        RET      ;vrat se
;
;
BARVY:  DB       81H, 0A1H, 0A1H
        DB       0A1H, 0A1H, 0A1H
        DB       0A1H, 71H
;
;
        END

```

Ing. Martin Štěpánek, J. Jovkova 3256, Praha 4

Multiprocesový operační systém ZX MULTITASKING pro počítače ZX Spectrum umožňuje existenci více programů v počítači a může zabezpečit jejich současný běh. Jedná se tedy o nové možnosti používání počítače, s jakými jsme se doposud setkávali jen u větších počítačů. Operační systém je navržen tak, že pracuje se všemi druhy programů (BASIC, strojový kód, Pascal atp.), pouze se dvěma omezeními: maximální velikost jednoho programu je 38016 bajtů místo 48 kB (musí totiž zůstat ještě místo na druhý proces — minimálně 10368 bajtů a prostor pro operační systém) a uživatelským procesům nejsou povoleny některé akce, které se týkají režimu přerušení procesoru.

V systému ZX MULTITASKING může například provozovat, ladit nebo vytvářet program na počítači, který počítá zároveň jiný program (využití čekacího času při dlouhotrvajících výpočtech). Další možností je využívání různých pomocných programů při práci s hlavním programem. Jako pomocný program můžeme mít například kalkulátor (viz příloha — program KALKULÁTOR/ZÁPISNÍK), kde si provádíme pomocné výpočty a poznámky, aniž bychom ovlivňovali hlavní program. Velmi výhodnou možností je využití multitaskingu pro elektronický diář, kde jako paralelní proces běží diář, který nás v době schůzek, vzkazů a jiných událostí může upozornit akustickým signálem a vypsát zprávu na monitor.

Podstatným rysem operačního systému ZX MULTITASKING je, že uživa-

tel navrhuje programy běžným způsobem, vytvořené programy jsou zcela normální (bez úprav) a fungují i na počítači bez tohoto operačního systému!

Operační systém ZX MULTITASKING může pracovat se ZX Microdrive, ZX Interface I, joysticky, tiskárnami a dalšími perifériemi bez omezení.

ZÁKLADNÍ POJMY

Abychom plně využili možností popísaného operačního systému, musíme si nejprve něco povědět o základních pojmech, se kterými se budeme během práce setkávat.

PROCES — je sled činností prováděných s určitými daty. Procesem zde tedy rozumíme nejen běh vlastního aplikačního programu, ale například i jeho editování, vypisování listingu apod., kdy vlastně běží určitá část operačního systému v ROM a operuje s daty, kterými je zde náš aplikační program.

MULTITASKING — je označení pro pseudoparalelní zpracování více procesů. Z hlediska uživatele se zdá, že procesy probíhají zároveň, ve skutečnosti však procesor střídavě pracuje buď pro jeden, nebo pro druhý proces.

PŘIDĚLENÍ PROCESORU PROCESU — znamená, že procesor je plně k dispozici potřebám procesu. Často říkáme, že proces je obsluhován.

AKTIVNÍ PROCES — je proces, kterému je trvale přidělen nebo automaticky střídavě přidělován procesor. V konkrétním okamžiku tedy může, ale též nemusí, být procesu přidělen procesor. Pokud je aktivních procesů více, je jasné, že procesor může být v každém okamžiku přidělen pouze jednomu z nich.

PASÍVNÍ PROCES — je takový proces, kterému trvale není přidělován procesor. Proces je jakoby zmrazen v určité své fázi a nevyvíjí se, dokud se nestane opět aktivním.

PROCES NA POPŘEDÍ — je procesem, který má neustále přístup ke klávesnici, a proto jen s ním můžeme komunikovat. Na popředí je tedy v dané době hlavní proces. Dále platí, že na popředí je vždy jen jeden proces.

PROCES NA POZADÍ — nemá přístup ke klávesnici standardní rutinou z ROM, takže se může pouze vykonávat, ale nemůže komunikovat. Tento fakt pozadí je nezbytný, protože, kdyby mělo popředí i pozadí přístup ke klávesnici, nedalo by se při stisku klávesy vlastně určit, ke kterému procesu stisk patří (protože oba běží paralelně). Pokud ovšem potřebujeme s procesem na pozadí komunikovat, operační systém nám umožní záměnu popředí s pozadím.

UCHOVÁVÁNÍ OBRAZOVKY PROCESU — je zachování obrazové informace příslušné procesu nezávisle na procesech ostatních. Obrazovku mohou uchovávat oba procesy, nebo jen proces na popředí, případně žádný z procesů.

ZPŮSOB ZAVEDENÍ SYSTÉMU

Do prázdného počítače nahrajeme program zápisem povelu **LOAD "zx-multi"**. Program se sám spustí a dotáže se nás na některé základní parametry. Základní hodnota bývá uvedena v závorce za otázkou a pokud s ní souhlasíme, nic nezadáme a stiskneme jen ENTER. Stejně se zachováváme, pokud na otázku nejsme schopni odpovědět.

První dotaz se týká odděleného uchovávání obrázku. Část paměti určená pro zápis informace, která se zobrazuje na monitoru, se totiž může, ale též nemusí, uchovávat. Pokud se uchovává, má každý proces svou obrazovku. Pokud se neuchovává, mají sice procesy obrazovku společnou, ale mohou se výrazně zlepšit časové charakteristiky běhu obou procesů a je k dispozici více paměti. Proto, pokud společná obrazovka není na závalu, volíme raději tento režim. Následující dotaz se týká rozdělení paměti mezi procesy. Rozdělení pracovní oblasti se dosáhne zadáním délky delšího procesu, která je normálně 31104 bajtů (bez obrazovky) při neuchovávání obrazovky a 24192 bajtů (bez obrazovky) při uchovávání obrazovky. Návoděda nám říká, jak dlouhý má být delší proces, pokud je kratším procesem program 16 kB. K rozdělení je nutné ještě podotknout, že proces nemůže mít libovolnou délku, ale lze ji nastavovat s krokem 3456 bajtů, což je jedna pracovní stránka (úprava zadané hodnoty na možnou se děje automaticky). Při rozdělení je též vhodné se snažit o maximální rozdíl délek obou procesů. Tehdy je totiž multitasking nejefektivnější a procesy běží nejrychleji. Posledním dotazem inicializačního programu je doba, po kterou má být proces přidělen po stisku klávesy. Pokud jsou totiž oba procesy aktivní, přiděluje se jim procesor střídavě. Klávesnice přitom patří procesu na popředí. Při stisku klávesy ovšem může být procesor právě přidělen procesu na pozadí a musí dojít k okamžitému přepnutí procesu. Aby při souvislé práci s popředím nebylo nutné znovu

a znovu provádět toto zpětné přidělování, zůstane procesor bezprostředně po stisku klávesy přidělen popředí na určitou dobu (právě zde zadávanou). Tato vlastnost je zvláště důležitá při režimu zachovávání obrazovek, protože jinak by nám při práci s popředím mohla rušivě problikávat obrazovka pozadí.

Po zodpovězení otázek se provede inicializace a vytvoření dvou prázdných procesů. K dispozici je nejdříve menší z procesů (tj. je na popředí, je mu přidělen procesor a je aktivní, zatímco druhý proces je pasivní). Nyní tedy můžeme zapsat — nebo nahrát pomocí **LOAD** — menší proces a poté přejít přes menu operačního systému (příkaz **VYMĚN**) k druhému, delšímu procesu.

POPIS FUNKCÍ SYSTÉMU

Vlastní operační systém ZX MULTITASKING běží na úplném pozadí dvou uživatelských procesů, ke kterým se může chovat různým způsobem. Režim operačního systému můžeme měnit příkazy z jeho menu, které se nám objeví při současném stisku kláves **SYMBOL SHIFT** a **SPACE**. Po celou dobu zobrazování menu jsou uživatelské procesy pasivní. Menu je rozděleno na dvě části: příkazovou (vlevo) a stavovou (vpravo).

Stavová část nás informuje o režimu operačního systému a procesu. Čtveřiček je vždy vyznačen proces, který je v souladu s nápisem. V prvním řádku vidíme délku obou procesů ve stránkách (1 str = 3456 bajtů) a blikáním je označen proces, který je na popředí. Ve druhém řádku nalezneme, který z procesů uchovává obrazovku. V dalším řádku je informace o aktivitě procesů a o řádek níže zpráva o právě přiděleném procesu.

Příkazová část obsahuje nápovědu způsobu volání dále uvedených příkazů (příkaz se vždy volá stiskem klávesy odpovídající inverzně zobrazenému písmenu):

VÝMĚNA PROCESŮ — způsobí okamžitou záměnu procesu na popředí s procesem na pozadí. Pokud je aktivní jen jeden proces, prohodí se přirozeně také tato aktivita, takže původní pasivní proces na pozadí se stane aktivním procesem na popředí. Jestliže se neuchovává obrazovka, musíme počítat s tím, že po přepnutí na druhý proces zůstane na monitoru stejná obrazovka, takže by mohlo dojít k nesprávné domněnce, že k přepnutí vůbec nedošlo.

1 PROCES AKTIVNÍ — aktivním se stane právě přidělený proces a druhý proces je pasivní. Aktivní proces bude zároveň na popředí.

2 PROCESY AKTIVNÍ — oba dva procesy se stanou aktivními a začne jim být střídavě přidělován procesor. Chování počítače při tomto režimu je různé, v závislosti na tom, uchováváme-li obrazovky. Pokud se uchovávají obrazovky obou procesů, dojde k jejich přepínání (projevuje se střídavým blikáním obrazovek na monitoru). Jestliže obrazovky nejsou uchovávány, zapisují oba procesy na společné stínítko a k blikání nedochází. Ve speciálním

případě může obrazovku uchovat jen jeden proces a tehdy střídavě vidíme vlastní obrazovku procesu s uchováváním a sloučenou obrazovku druhého procesu.

ZRUŠENÍ PROCESU — způsobí vymazání a inicializaci procesu, kterému je právě přidělen procesor. Je to náhrada za příkaz **NEW**, respektive **RAND USR 0**, z nichž ani jeden nemůžeme použít, protože by došlo k odpojení nebo zničení druhého procesu!

STOP PROGRAMU — zastaví běh programu v právě přiděleném procesu. Na rozdíl od klávesy **BREAK** (kterou se zde nedoporučuje používat) je tento příkaz schopný zastavit i strojový kód.

ENTER — stisk této klávesy opustí zobrazené menu bez změny režimu operačního systému.

Kromě základních příkazů uvedených v menu existují ještě další méně typické příkazy:

X — stiskem této klávesy provedeme zrušení obou procesů a úplnou inicializaci systému.

N — tato klávesa střídavě zapíná a vypíná režim uchovávání obrázku u pozadí (lze ji ovšem používat jen tehdy, pokud jsme při úvodní instalaci systému požadovali uchovávání obrazovky). Tuto funkci se však nedoporučuje příliš používat, protože vzhledem ke své zvláštní povaze může často vést k poškození procesů nebo ke zhroucení systému. Protože funkce mění rozdělení paměti, dochází k přesunům bloků paměti, což může mít za následek zobrazení nesmyslných znaků na obrazovce. Pokud k tomu dojde, není to závada a obrazovku příslušného procesu můžeme vymazat běžným příkazem **CLS**.

Po přečtení popisu příkazů bez praktických zkušeností se může zdát jejich používání složité. Pokud si ale vše vyzkoušíte v praxi, zjistíte, že složitost je pouze zdánlivá a že změny chování v důsledku příkazů jsou zcela logické a očekávatelné.

PRINCIP ČINNOSTI

Úkolem operačního systému ZX MULTITASKING je umožnění existence a chodu více programů v počítači současně. Protože ve skutečnosti procesor nemůže obsluhovat více programů zároveň, je paralelní obsluha procesů simulována rychlým přepínáním procesoru mezi procesy. Při každém přepnutí procesoru je ovšem nezbytné zapamatovat si přesně stav procesu v okamžiku jeho zastavení a uložit veškeré informace v registrech procesoru. Při zpětném přepnutí procesoru je činnost právě opačná. Registry procesoru se naplní původními hodnotami a proces pokračuje přesně v tom stavu, v jakém byl opuštěn. Přepínání mezi procesy se děje automaticky. Přepínací program je aktivován periodicky podle

přerušování procesoru vnitřními hodinami. Protože předání procesoru mezi procesy nenastává okamžitě, ale trvá určitou dobu, je doba, po kterou je procesor přidělen jednomu procesu, stanovena tak, aby poměr ztrátových časů přepínání k časům běhu procesů byl přijatelný. V našem případě je procesor přidělován procesu na dobu 100 ms. Doba přepnutí kolísá podle režimu, takže v nejhroším případě je ztrátový čas procesoru asi 50 % z celkového strojového času a v nejlepším případě 14 %.

Přidělování procesoru se řídí následujícími zásadami. Pokud je jeden z procesů pasivní, je procesor trvale přidělen procesu aktivnímu. Jsou-li oba procesy aktivní, je procesor střídavě přidělován oběma procesům. Přidělování je periodické a na stejnou dobu. Jestliže ale v tomto režimu stiskneme klávesu, je procesor okamžitě přidělen procesu na popředí a v následujícím definovaném čase (mnohem delším než základní čas přidělení procesoru) zůstane procesor přidělen „popředí“, čímž se zamezí zbytečným opakování téže situace při komunikaci uživatele s programem a zkrátí se odezva procesu na stisk klávesy. Poslední možností přidělování je nedobrovolné narušení režimu rozdělování strojového času vlivem činnosti uživatelského programu. Pokud totiž uživatelský program zakáže přerušování (dělají to například příkazy BEEP, SAVE, LOAD a další), zůstane procesor trvale přidělen tomuto procesu, až do povolení přerušování.

Počítač ZX Spectrum není přizpůsoben na paralelní zpracování dat. Jeho systémové proměnné jsou například na pevných adresách a nikoli pod ukazatelem, což znamená, že systémové proměnné musí být uloženy. Podobná situace je s obsahem obrazovky, který je vždy od adresy 16384. Také programy jsou většinou vázány na konkrétní absolutní adresy a musí se proto v rámci přepínání procesu celé přesunovat.

Z dosud uvedeného vyplývá, že nezáhame-li k technickým úpravám počítače, je jediná možnost realizace více procesů jejich přenášení mezi pracovní a skladovací pozici v operační paměti. Algoritmus je přitom navržen tak, že se nepřetáhají celé procesy, ale jenom část paměti rovná délce kratšího procesu. Proto má také operační systém největší účinnost při maximálním rozdílu délek procesů a neuchovávání obrazovky, kdy se přenáší pouze blok paměti v délce jedné stránky (stránkování bylo vynuceno právě nutností realizace jednoduchého a rychlého algoritmu přepínání procesů).

PŘÍKLAD POUŽITÍ

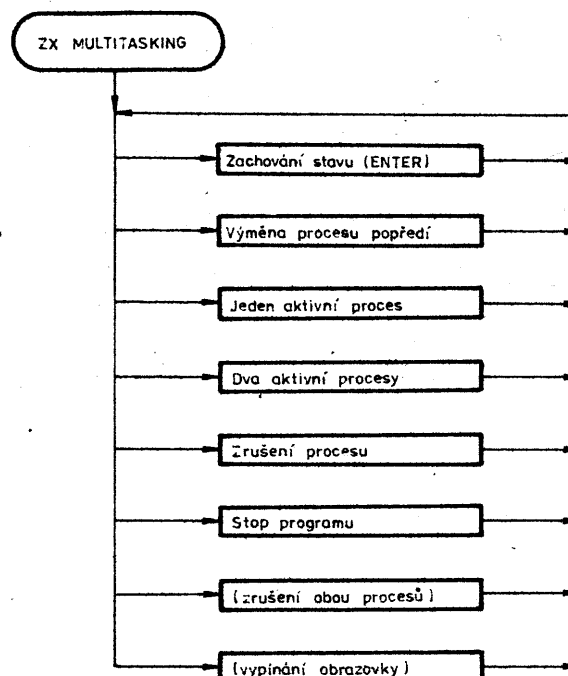
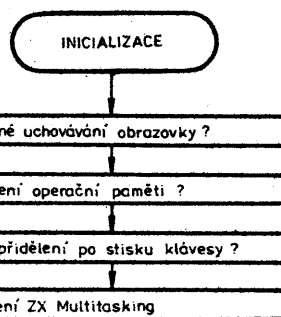
PROGRAM KALKULÁTOR/ZÁPISNÍK

Program KALKULÁTOR/ZÁPISNÍK je ukázkou využívání výhod paralelních programů pod systémem ZX MULTITASKING. Program, jak název napovídá, umožňuje provádění běžných aritmetických výpočtů a zapisování výsledků a jiných poznámek do zápisníkové části. Celý program je napsán v jazyku

BASIC a instalujeme ho na pozadí hlavního procesu, kterým bude střídavě podle potřeby kalkulátor nebo hlavní program uživatele.

Nahrání kalkulátoru provedeme následovně. Instalujeme operační systém ZX MULTITASKING. Uchovávání obrazovky raději požadujeme (abychom pro první přiblížení neměli problémy s orientací v procesech) a delší proces necháme v maximální délce, což je délka naznačená v závorce. Po inicializaci systému se nacházíme v menším procesu a tak můžeme přímo nahrát program KALKULÁTOR/ZÁPISNÍK příkazem LOAD „kalkulátor“. Poté můžeme příkazem VYMĚN v menu operačního systému (zavolá se současným stiskem SYMBOL SHIFT a SPACE) přejít k druhému procesu a pracovat na něm. Když si potřebujeme něco vypočítat, zapsat nebo naopak ze zápisníku přečíst, přepneme opět příkazem VYMĚN na kalkulátor.

Abychom mohli s kalkulátorem pracovat, uvedeme si zde přehled jeho příkazů. Předně je nutné si uvědomit, že program pracuje ve dvou režimech (kalkulátor a zápisník), mezi kterými přepínáme stiskem ENTER. Po prvním spuštění programu se vždy nacházíme v režimu kalkulátoru (nadále se pozná podle zobrazení názvu programu v první řádce). Kalkulátor očekává zapsání aritmetického výrazu, který má vypočítat. Zapišeme-li tedy například 12*56, zobrazí se výsledek 672, který se zároveň uloží do výsledkové proměnné „v“. Ve výsledkové proměnné se vždy



nachází poslední výsledek. Výsledkovou proměnnou můžeme užít v následujícím aritmetickém výrazu (např. v/2 dá nyní výsledek 336). Zadávat můžeme libovolné aritmetické výrazy povolené syntaxí BASICu, tedy obsahující i funkce a relace. Nemůžeme použít proměnných ve výrazu, kromě výsledkové proměnné „v“. Pokud chceme poslední výsledek zapsat do zápisníku, zapišeme místo výrazu „t“. Když si chceme udělat v zápisníku jinou poznámku, uvedeme její text znakem „f“ (např. \$Kčs/rok). Dojde-li při výpočtu k chybě a přerušeni programu, pokračujeme příkazem GOTO 2!

Do režimu zápisníku se dostaneme z kalkulátoru stiskem ENTER. V zápisníku můžeme vybírat jednotlivé řádky kurzory „nahoru“ a „dolů“. Nový text do řádky vložíme stiskem „t“ a zapsáním textu. Stiskem „f“ přeneseme samostatnou hodnotu ve vybraném řádku do výsledkové proměnné „v“. Stiskem ENTER se dostaneme zpět do kalkulátoru.

K ověření také můžeme použít programy „p1“ a „p2“, které nahrajeme každým do jednoho procesu, oba je spustíme a režim systému nastavíme na dva aktivní procesy. Velmi názorně tak uvidíme, jak systém střídavě přiděluje procesor. Do většího procesu (o dostatečné velikosti!!!) také můžeme nahrát nějakou hru pro 16 kB Spectrum, například „mazeman“, na níž si můžeme vyzkoušet i zastavení jejího strojového programu funkcí STOP v operačním systému.

Vypis programu ZX—Multitasking

```

1 CLEAR 29999: LOAD ""CODE
5 INK 1: PAPER 6: BORDER 7
10 CLS : PRINT "ZX MULTITASKING * Stepanek '87"

11 PRINT AT 20,0; PAPER 5;"ENTER-uzije hodnotu v zavorce !!!":
PRINT "NEVITE-LI STISKNETE 'ENTER' !"
12 PRINT AT 3,0;"Oddelene uchovavani obr.: (NE)": INPUT "Ano/N
e? ";o$
14 LET spodek=23296: IF o$="a" THEN LET spodek=16384: PRINT A
T 3,26;"ANO "
16 PRINT : PRINT "ROZDELENI PAMETI PRO DVA PROCESY"
18 PRINT : PRINT "DELSI PROCES BEZ SCR=(31104)": PRINT PAPER
5;AT 12,0;"(Pro 'mensi'=16KB zadejte 24000)": IF o$="a" THEN PR
INT AT 7,22;"24192";AT 12,26; PAPER 5;"17000"
19 INPUT "Kolik bytes? ";v$: IF v$="" THEN LET sv=(o$="a")*7+
(o$<"a")*9: GO TO 22
20 LET sv=INT (0.99999+VAL (v$)/3456): IF sv>9 OR sv>7 AND o$=
"a" THEN PRINT #0;AT 1,0;"PRILIS VELKY !": BEEP 2,30: GO TO 19
22 LET v=23295+3456*sv
24 LET sm=10-sv-2*(o$="a"): LET m=23295+3456*sm
25 IF sm>sv THEN PRINT #0;AT 1,0;"PROCES NENI VETSI !": BEEP
2,30: GO TO 19
26 PRINT AT 7,21;sv*3456;" bytes";AT 8,14;"se scr=";sv*3456+69
12;" bytes"
28 PRINT "--konec pridelene RAM=";v,"--hodnota RAMTOP      =" ;v-1
68
30 PRINT : PRINT "KRAT. PROCES BEZ SCR=";sm*3456;" bytes";AT 1
3,14;"se scr=";sm*3456+6912;" bytes"
32 PRINT "--konec pridelene RAM=";m,"--hodnota RAMTOP      =" ;m-1
68
34 PRINT : PRINT "PRIDELENI PO STISKU klav.=(2s)": INPUT "Koli
k sekund? ";p$: IF p$="" THEN LET p=100: GO TO 40
36 LET p=50*VAL p$: IF p>200 THEN LET p=200
40 POKE min,spodek-256*INT (spodek/256): POKE min+1,INT (spode
k/256): POKE vetsi,v-256*INT (v/256): POKE vetsi+1,INT (v/256):
POKE mensi,m-256*INT (m/256): POKE mensi+1,(m/256): POKE vydrz,p
50 CLS : PRINT AT 0,0; PAPER 6; INK 1;"ZX MULTITASKING v1.0 St
epanek'87 POVELY:      POPREDI(str): ";sm;" ";sv;" v-vymena
Obr.uchovava:      1-1 aktivni Aktivni proc:      2-2 aktivni
Prideleny pr:      z-zrus proc Basic zprava:      CS-stop prg
ENTER-uchova stav
52 PLOT 0,167: DRAW 0,-55: DRAW 255,0: DRAW 0,55: PLOT 105,167
: DRAW 0,-54
99 RANDOMIZE USR startb

```

```

L
00020 ;*****
00040 ;* ZX MULTITASKING *
00060 ;*****
00080 ;
00100 ; c Martin STEPANEK 1987
00120 ;
00140 BASIC EQU #12A9 ;MAIN-1
00160 PRAMT EQU 23732
00180 RAMTOP EQU 23730
00200 UDG EQU 23675
00220 RASP EQU 23608
00240 CHARS EQU 23606
00260 ERRSP EQU 23613
00280 CHANS EQU 23631
00300 DATADD EQU 23639
00320 PROG EQU 23635
00340 VARS EQU 23627
00360 ELINE EQU 23641
00380 WORKSP EQU 23649
00400 STKBOT EQU 23651
00420 STKEND EQU 23653
00440 ATTRP EQU 23693
00460 ATTRT EQU 23695
00480 BORDCR EQU 23624
00500 REPDEL EQU 23561
00520 FLAGS EQU 23611
00540 ERRNR EQU 23610
00560 ;=====
00580 ORG 63743
00600 DISP 1-23744+5000
00620 ;-----
00640 ADRESA DEFW RIZENI;PRCSU
00660 N1 DEFW 0;UK. DOLNIHO
00680 N2 DEFW 0;UK. HORNIHO
00700 MIN DEFW 23296;BEZ SCR
00720 SP1 DEFW 0;SP PROCES 1
00740 SP2 DEFW 0;SP PROCES 2
00760 NAVRAT DEFW 0;NAVRAT. ADR
00780 VETSI DEFW 0;PROCES(MEZ)
00800 MENSI DEFW 0;PROCES(MEZ)

```

```

00820 MEZ DEFW 0;MEZI PROC.
00840 TEXT DEFW 0;UVODNI INFO
00860 AKTPROC DEFB 1;AKT.PROCES
00880 POPREDI DEFB 1;KLAV&ZOBR
00900 POCET DEFB 0;PRUCHODY
00920 STOPB DEFB 0;PRO BREAK
00940 OBA DEFB 0;PRC. ZAROVEN
00960 VYDRZ DEFB 120;PO STISKU
00980 MENU DEFB 0;FLAG ZOBRAZ
01000 TEXTV DEFB 32+128
01020 DEFM /* Vetsi pro/
01040 DEFM /ces /
01060 TEXTM DEFB 32+128
01080 DEFM /* Mensi pro/
01100 DEFM /ces /
01120 TEXTA DEFB 32+128
01140 DEFM /
01160 DEFM /
01180 DEFM /
01200 DEFM /
01220 DEFM /
01240 DEFB 127;COPYRIGHT
01260 DEFM / 1987 Marti/
01280 DEFM /n Stepanek /
01300 DEFB 32+128
01320 NEZOBR DEFW 0;POZADI NEZO
01340 MSG1 DEFB 0;BASIC1 MSG+
01360 MSG2 DEFB 0;BASIC2 MSG+
01380 ;-----
01400 START DI
01420 LD HL,54399
01440 LD (VETSI),HL;PROC
01460 LD HL,26751
01480 LD (MENSI),HL;PROC
01500 LD HL,23296;SPODEK
01520 LD (MIN),HL
01540 JP ZAVED
01560 ;-----
01580 STARTX DI
01600 CALL VYMEN
01620 JP ZAVED

```

```

01640 ;-----
01660 STOP IM 1
01680 RET
01700 ;-----
01720 RIZENI DI ; PARALELNICH
01740 PUSH AF ; PROCESU
01760 LD A,(STOPB)
01780 CP 1
01800 JP Z,BREAK
01820 XOR A ;CELA KLAVES
01840 IN A,(254)
01860 AND 31
01880 CP 31
01900 JP NZ,STISK
01920 LD A,(OBA);ZAROVEN
01940 CP 0
01960 JP Z,ROMKEY
01980 LD A,(POCET)
02000 INC A
02020 LD (POCET),A
02040 CP 10;MEZ PREPNUTI
02060 JP NZ,BEZKEY
02080 XOR A
02100 LD (POCET),A
02120 POP AF
02140 JP PROCES ;VYMEN
02160 ;---
02180 ROMKEY POP AF
02200 JP 56 ;POKR. V ROM
02220 ;---
02240 BEZKEY POP AF
02260 EI
02280 RET ;POKR. BEZ KEY
02300 ;-----
02320 BREAK XOR A
02340 LD (STOPB),A
02360 LD A,(FLAGS)
02380 RES 7,A
02400 LD (FLAGS),A
02420 LD A,150
02440 LD (POCET),A
02460 POP AF
02480 EI
02500 JP BASIC
02520 ;-----
02540 STISK PUSH BC
02560 LD BC,#7FFE ;"OS"
02580 IN A,(C)
02600 AND 31
02620 CP 28;"SPACE"+"SS"
02640 JP Z,MENUON
02660 LD A,(VYDRZ)
02680 LD B,A
02700 LD A,255
02720 SUB B
02740 LD (POCET),A;ZPOZD
02760 LD A,(AKTPROC)
02780 LD B,A
02800 LD A,(POPREDI)
02820 CP B
02840 POP BC
02860 JP Z,ROMKEY
02880 POP AF
02900 JP PROCES ;VYMEN
02920 ;-----
02940 POVELY PUSH BC
02960 POV LD BC,#BFFE
02980 IN A,(C)
03000 AND 31
03020 CP 30;"ENTER"
03040 JP NZ,POV1
03060 POV0 CALL MENUOFF
03080 JP POVON
03100 ;---
03120 POV1 LD BC,#F7FE
03140 IN A,(C)
03160 AND 31
03180 CP 30;"1"PROCES
03200 JP NZ,POV2
03220 CALL MENUOFF
03240 XOR A
03260 LD (OBA),A
03280 POV11 LD A,(POPREDI)
03300 LD B,A
03320 LD A,(AKTPROC)
03340 CP B

```

| | | | | | |
|-------|---------------------------|-------|---------------------------|-------|--------------------------|
| 03360 | JP Z,POVKON | 05060 | LD A,1+40+64;BARVA | 06880 | XOR A |
| 03380 | POP BC | 05080 | LD (22620),A ;SCR+ | 06900 | CP B |
| 03400 | POP AF | 05100 | LD (22622),A ;SCR+ | 06920 | JP NZ,OKNO |
| 03420 | JP PROCES ;VYMEN | 05120 | JP STAVY5 | 06940 | CP C |
| 03440 | ---- | 05140 | ---- | 06960 | JP NZ,OKNO |
| 03460 | POV2 CP 29 ;"2"PROCESY | 05160 | STAVY3 LD A,(POPREDI) | 06980 | LD HL,22528;ZAC.AT |
| 03480 | JP NZ,POV3 | 05180 | CP 1 | 07000 | LD DE,61312+2048 |
| 03500 | INC A ;TJ.A=1 | 05200 | LD A,1+40+64;BARVA | 07020 | LD B,0 |
| 03520 | LD (OBA),A | 05220 | JP Z,STAVY4 | 07040 | ATRIB LD A,(HL) |
| 03540 | JP POVO | 05240 | LD (22622),A ;SCR+ | 07060 | PUSH AF |
| 03560 | ---- | 05260 | JP STAVY5 | 07080 | LD A,(DE) |
| 03580 | POV3 LD BC,#FEFE | 05280 | ---- | 07100 | LD (HL),A |
| 03600 | IN A,(C) | 05300 | STAVY4 LD (22620),A ;SCR+ | 07120 | POP AF |
| 03620 | AND 31 | 05320 | STAVY5 LD A,(OBA) | 07140 | LD (DE),A |
| 03640 | CP 15 ;"V"VYMEN | 05340 | CP 0 | 07160 | INC HL |
| 03660 | JP NZ,POV4 | 05360 | JP Z,STAVY6 | 07180 | INC DE |
| 03680 | CALL MENUOFF | 05380 | LD A,1+32+64;BARVA | 07200 | DJNZ ATRIB |
| 03700 | LD A,(POPREDI) | 05400 | LD (22652),A ;AKT+ | 07220 | POP BC |
| 03720 | INC A | 05420 | LD (22654),A ;AKT+ | 07240 | POP DE |
| 03740 | LD (POPREDI),A | 05440 | JP STAVY8 | 07260 | POP HL |
| 03760 | CP 3 | 05460 | ---- | 07280 | RET |
| 03780 | JP NZ,POV31 | 05480 | STAVY6 LD A,(AKTPROC) | 07300 | ---- |
| 03800 | LD A,1 | 05500 | CP 1 | 07320 | ZAVED DI ; NOVE PROCESY |
| 03820 | LD (POPREDI),A | 05520 | LD A,1+32+64;BARVA | 07340 | CALL STOP |
| 03840 | POV31 POP BC | 05540 | JP Z,STAVY7 | 07360 | LD A,1 |
| 03860 | POP AF | 05560 | LD (22654),A ;AKT+ | 07380 | LD (POPREDI),A |
| 03880 | JP PROCES ;VYMEN | 05580 | JP STAVY8 | 07400 | LD A,0 |
| 03900 | ---- | 05600 | ---- | 07420 | LD (NEZOBR+1),A |
| 03920 | POV4 CP 29 ;"Z"ZRUS | 05620 | STAVY7 LD (22652),A ;AKT+ | 07440 | LD (NEZOBR),A |
| 03940 | JP Z,NEW | 05640 | STAVY8 LD A,(AKTPROC) | 07460 | LD (STOPX),A |
| 03960 | CP 27 ;"X"ZAVED | 05660 | CP 1 | 07480 | LD (POCET),A |
| 03980 | JP Z,STARTX | 05680 | LD A,1+40+64;BARVA | 07500 | LD (OBA),A |
| 04000 | CP 30 ;"CS"BREAK | 05700 | JP Z,STAVY9 | 07520 | LD (MSG1),A |
| 04020 | JP NZ,POV6 | 05720 | LD (22686),A;PRID+ | 07540 | LD (MSG2),A |
| 04040 | LD A,1 | 05740 | JP STAV10 | 07560 | LD (MENU),A |
| 04060 | LD (STOPB),A | 05760 | ---- | 07580 | LD HL,(VETSI);PROC |
| 04080 | CALL MENUOFF | 05780 | STAVY9 LD (22684),A;PRID+ | 07600 | LD (MEZ),HL |
| 04100 | JP POV11 | 05800 | STAV10 LD A,(AKTPROC) | 07620 | LD HL,TEXTV |
| 04120 | ---- | 05820 | CP 1 | 07640 | LD (TEXT),HL |
| 04140 | POV6 LD BC,#7FFE | 05840 | LD A,(ERRNR) | 07660 | CALL INIT |
| 04160 | IN A,(C) | 05860 | JP Z,STAV11 | 07680 | JR NOVYP2 |
| 04180 | AND 31 | 05880 | CP 255 ; BEZ ERR | 07700 | STDAL LD HL,(MENSI);PROC |
| 04200 | CP 23 ;"N"NEZOBRZ | 05900 | LD A,0 | 07720 | LD (MEZ),HL |
| 04220 | JP NZ,POV | 05920 | LD (MSG2),A ; MSG- | 07740 | LD HL,TEXTM |
| 04240 | LD A,(NEZOBR+1) | 05940 | JP Z,STAV12 | 07760 | LD (TEXT),HL |
| 04260 | LD BC,0 | 05960 | LD A,1 | 07780 | STDAL2 CALL INIT |
| 04280 | LD (NEZOBR),BC | 05980 | LD (MSG2),A ; MSG+ | 07800 | LD A,248 |
| 04300 | CP 0 | 06000 | JP STAV12 | 07820 | LD I,A ;ADR=63743 |
| 04320 | JP NZ,POVO | 06020 | ---- | 07840 | IM 2 |
| 04340 | LD BC,6912 | 06040 | STAV11 CP 255 ; BEZ ERR | 07860 | EI |
| 04360 | LD (NEZOBR),BC | 06060 | LD A,0 | 07880 | RET |
| 04380 | JP POVO | 06080 | LD (MSG1),A ; MSG- | 07900 | ---- |
| 04400 | POVKON POP BC | 06100 | JP Z,STAV12 | 07910 | NEW DI ; AKTUALNI PROC |
| 04420 | POP AF | 06120 | LD A,1 | 07920 | CALL MENUOFF |
| 04440 | EI | 06140 | LD (MSG1),A ; MSG+ | 07940 | LD HL,(PRAMT) |
| 04460 | RET | 06160 | STAV12 CALL MSGX | 07960 | LD (MEZ),HL |
| 04480 | ---- | 06180 | STAV14 POP HL | 07980 | LD HL,TEXTV |
| 04500 | MENUON LD A,(MENU) | 06200 | JP POV | 08000 | LD (TEXT),HL |
| 04520 | CP 1 | 06220 | ---- | 08020 | LD A,(AKTPROC) |
| 04540 | JP Z,POV | 06240 | MSGX LD A,(MSG1) | 08040 | CP 2 |
| 04560 | LD A,1 | 06260 | CP 0 | 08060 | JP Z,STDAL2 |
| 04580 | LD (MENU),A | 06280 | JP Z,MSGX2 ;MSG1- | 08080 | LD HL,TEXTM |
| 04600 | CALL VYMEN ;OKENKO | 06300 | LD A,1+32+64;BARVA | 08100 | LD (TEXT),HL |
| 04620 | PUSH HL | 06320 | LD (22716),A;MSG1+ | 08120 | JP STDAL2 |
| 04640 | LD A,1+48+64;BARVA | 06340 | LD A,(MSG2) | 08140 | ---- |
| 04660 | LD B,0;DELKA 256 B | 06360 | CP 0 | 08160 | NOVYP2 DI ; TYP AMINUS |
| 04680 | LD HL,22528;ZAC.AT | 06380 | RET Z ;MSG2- | 08180 | LD IV,BASIC;NAVRAT |
| 04700 | ZAKLAD LD (HL),A | 06400 | LD A,1+32+64;BARVA | 08200 | PUSH IV ; ADRESA |
| 04720 | INC HL | 06420 | LD (22718),A;MSG2+ | 08220 | LD IV,#5C3A |
| 04740 | DJNZ ZAKLAD | 06440 | RET | 08240 | PUSH AF ;SIMULACE |
| 04760 | STAVY0 LD A,(POPREDI) | 06460 | ---- | 08260 | PUSH BC ;ULOZENI |
| 04780 | CP 1 | 06480 | MENUOFF XOR A | 08280 | PUSH DE ;REGISTRU |
| 04800 | LD A,1+48+64+128 | 06500 | LD (MENU),A | 08300 | PUSH HL |
| 04820 | JP Z,STAVY1 | 06520 | CALL VYMEN ;OKENKO | 08320 | EXX |
| 04840 | LD (22590),A ;POPR | 06540 | RET | 08340 | EX AF,AF' |
| 04860 | JP STAVY2 | 06560 | ---- | 08360 | PUSH AF |
| 04880 | ---- | 06580 | VYMEN PUSH HL ;OKENKO | 08380 | PUSH BC |
| 04900 | STAVY1 LD (22588),A ;POPR | 06600 | PUSH DE | 08400 | PUSH DE |
| 04920 | STAVY2 LD HL,(MIN) | 06620 | PUSH BC | 08420 | PUSH HL |
| 04940 | LD A,H | 06640 | LD HL,16384;OKNO | 08440 | PUSH IX |
| 04960 | CP 64 ;=KDYZ 16384 | 06660 | LD DE,61312;BUFFER | 08460 | PUSH IY |
| 04980 | JP NZ,STAVY5 ;SCR- | 06680 | LD BC,2048 ;DELKA | 08480 | LD (SP2),SP |
| 05000 | LD A,(NEZOBR+1) | 06700 | LD A,(HL) | 08500 | LD BC,3456 |
| 05020 | CP 0 ; = 0 PRO OBA | 06720 | PUSH AF | 08520 | LD DE,(MIN) |
| 05040 | JP NZ,STAVY3 | 06740 | LD A,(DE) | 08540 | LD HL,(PRAMT) |
| | | 06760 | LD (HL),A | 08560 | ADD HL,BC |
| | | 06780 | POP AF | 08580 | INC HL |
| | | 06800 | LD (DE),A | 08600 | NP2AM1 LD (N1),DE |
| | | 06820 | INC HL | 08620 | LD (N2),HL |
| | | 06840 | INC DE | 08640 | SCF |
| | | 06860 | DEC BC | 08660 | CCF |

| | | | | | |
|-------|-----------------------------|-------|------------------------|-------|----------------------|
| 08680 | SBC HL, BC | 10500 | JP AP1 | 12320 | LD BC, 16383 |
| 08700 | EX DE, HL | 10520 | ----- | 12340 | SBC HL, BC |
| 08720 | LDIR | 10540 | AMINUS LD (SP2), SP | 12360 | LD B, H |
| 08740 | LD A, (N2+1) | 10560 | LD A, (NEZOBR+1) | 12380 | LD C, L |
| 08760 | CP 226 ; HI 57856 | 10580 | CP 0 | 12400 | LD HL, (MEZ) |
| 08780 | JP Z, NP2AM3 | 10600 | CALL NZ, SCRMS | 12420 | POP DE ; NAVRATOVA |
| 08800 | NP2AM2 LD BC, 3456 | 10620 | LD BC, (NEZOBR) | 12440 | LD (NAVRAT), DE; ADR |
| 08820 | LD DE, (N2) | 10640 | LD HL, (MIN) | 12460 | LD (HL), A |
| 08840 | LD HL, (N1) | 10660 | ADD HL, BC | 12480 | DEC HL : DEC BC |
| 08860 | ADD HL, BC | 10680 | EX DE, HL | 12500 | CP B |
| 08880 | EX DE, HL | 10700 | LD HL, (PRAMT) | 12520 | JR NZ, NULUJ |
| 08900 | ADD HL, BC | 10720 | ADD HL, BC | 12540 | CP C |
| 08920 | JP NP2AM1 | 10740 | LD BC, 3456 | 12560 | JR NZ, NULUJ |
| 08940 | ----- | 10760 | ADD HL, BC | 12580 | LD HL, (MEZ) |
| 08960 | NP2AM3 LD A, 1; AKT. PROCES | 10780 | INC HL | 12600 | LD (PRAMT), HL |
| 08980 | LD (AKTPROC), A | 10800 | LD (N1), DE | 12620 | LD DE, #3EAF |
| 09000 | JP STDAL | 10820 | LD (N2), HL | 12640 | LD BC, #00A8 |
| 09020 | ----- | 10840 | SCF | 12660 | EX DE, HL |
| 09040 | PROCES DI; ULOZENI/NACTENI | 10860 | CCF | 12680 | LDDR ; PLNENI UDG |
| 09060 | PUSH AF | 10880 | SBC HL, BC | 12700 | EX DE, HL |
| 09080 | PUSH BC | 10900 | EX DE, HL | 12720 | INC HL |
| 09100 | PUSH DE | 10920 | LDIR | 12740 | LD (UDG), HL |
| 09120 | PUSH HL | 10940 | LD BC, 3456 ; DELKA | 12760 | DEC HL |
| 09140 | EXX | 10960 | LD DE, (N1) ; CIL | 12780 | LD BC, #0040 |
| 09160 | EX AF, AF' | 10980 | LD HL, (N2) ; ZDROJ | 12800 | LD (RASP), BC |
| 09180 | PUSH AF | 11000 | LDIR | 12820 | LD (RAMTOP), HL |
| 09200 | PUSH BC | 11020 | LD A, (N2+1) | 12840 | LD HL, #3C00 |
| 09220 | PUSH DE | 11040 | CP 226 ; HI 57856 | 12860 | LD (CHARS), HL |
| 09240 | PUSH HL | 11060 | JP NZ, AM2 | 12880 | LD HL, (RAMTOP) |
| 09260 | PUSH IX | 11080 | LD A, 1 | 12900 | LD (HL), #3E |
| 09280 | PUSH IY | 11100 | LD (AKTPROC), A | 12920 | DEC HL |
| 09300 | LD A, (AKTPROC) | 11120 | LD SP, (SP1) | 12940 | LD SP, HL |
| 09320 | CP 1 | 11140 | JP PRZPET | 12960 | DEC HL |
| 09340 | JP Z, APLUS | 11160 | ----- | 12980 | DEC HL |
| 09360 | JP AMINUS | 11180 | AM2 LD BC, 3456 | 13000 | LD (ERRSP), HL |
| 09380 | PRZPET POP IY | 11200 | LD DE, (N2) | 13020 | LD IY, #5C3A |
| 09400 | POP IX | 11220 | LD HL, (N1) | 13040 | LD HL, #5CBB |
| 09420 | POP HL | 11240 | ADD HL, BC | 13060 | LD (CHANS), HL |
| 09440 | POP DE | 11260 | EX DE, HL | 13080 | LD DE, #15AF |
| 09460 | POP BC | 11280 | ADD HL, BC | 13100 | LD BC, #0015 |
| 09480 | POP AF | 11300 | JP AM1 | 13120 | EX DE, HL |
| 09500 | EX AF, AF' | 11320 | ----- | 13140 | LDIR |
| 09520 | EXX | 11340 | SCRPLS LD A, (POPREDI) | 13160 | EX DE, HL |
| 09540 | POP HL | 11360 | CP 1 | 13180 | DEC HL |
| 09560 | POP DE | 11380 | JP Z, ULOZSC; SCREEN | 13200 | LD (DATADD), HL |
| 09580 | POP BC | 11400 | LD BC, 6912 | 13220 | INC HL |
| 09600 | POP AF | 11420 | LD HL, (N2) | 13240 | LD (PROG), HL |
| 09620 | EI | 11440 | SCF | 13260 | LD (VARS), HL |
| 09640 | RET | 11460 | CCF | 13280 | LD (HL), #80 |
| 09660 | ----- | 11480 | SBC HL, BC | 13300 | INC HL |
| 09680 | APLUS LD (SP1), SP | 11500 | LD DE, 16384 | 13320 | LD (ELINE), HL |
| 09700 | LD BC, 3456 ; BLOK | 11520 | LDIR | 13340 | LD (HL), #0D |
| 09720 | LD HL, (PRAMT) | 11540 | RET | 13360 | INC HL |
| 09740 | SCF | 11560 | ----- | 13380 | LD (HL), #80 |
| 09760 | CCF | 11580 | ULOZSC LD BC, 3456 | 13400 | INC HL |
| 09780 | SBC HL, BC | 11600 | LD HL, (N2) | 13420 | LD (WORKSP), HL |
| 09800 | INC HL | 11620 | SCF | 13440 | LD (STKBOT), HL |
| 09820 | EX DE, HL | 11640 | CCF | 13460 | LD (STKEND), HL |
| 09840 | LD HL, 54400 | 11660 | SBC HL, BC | 13480 | LD A, #38 |
| 09860 | AP1 LD (N1), DE | 11680 | EX DE, HL | 13500 | LD (ATTRP), A |
| 09880 | LD (N2), HL | 11700 | LD HL, 16384 | 13520 | LD (ATTRT), A |
| 09900 | ADD HL, BC | 11720 | LD BC, 6912 | 13540 | LD (BORDCR), A |
| 09920 | EX DE, HL | 11740 | LDIR | 13560 | LD HL, #0523 |
| 09940 | LDIR | 11760 | RET | 13580 | LD (REPDEL), HL |
| 09960 | LD HL, (N2) ; ZDROJ | 11780 | ----- | 13600 | DEC (IY-58) |
| 09980 | LD DE, (N1) ; CIL | 11800 | SCRMS LD A, (POPREDI) | 13620 | DEC (IY-54) |
| 10000 | LD BC, 3456 ; BLOK | 11820 | CP 2 | 13640 | LD HL, #15C6 |
| 10020 | LDIR | 11840 | JP Z, ULOZ ; SCREEN | 13660 | LD DE, #5C10 |
| 10040 | LD HL, (N1) | 11860 | LD BC, 3457 | 13680 | LD BC, #000E |
| 10060 | LD A, (NEZOBR+1) | 11880 | LD HL, (PRAMT) | 13700 | LDIR |
| 10080 | LD B, A | 11900 | ADD HL, BC | 13720 | SET 1, (IY+1) |
| 10100 | LD A, (MIN+1) | 11920 | LD BC, 6912 | 13740 | CALL #0EDF |
| 10120 | ADD A, B | 11940 | LD DE, 16384 | 13760 | LD (IY+49), 02 |
| 10140 | CP H | 11960 | LDIR | 13780 | CALL #0D6B |
| 10160 | JP NZ, AP2 | 11980 | RET | 13800 | XOR A |
| 10180 | LD A, B | 12000 | ----- | 13820 | LD DE, (TEXT) |
| 10200 | CP 0 | 12020 | ULOZ LD BC, 6912 | 13840 | CALL #0C0A |
| 10220 | CALL NZ, SCRPLS | 12040 | LD DE, (PRAMT) | 13860 | XOR A |
| 10240 | LD A, 2 | 12060 | LD HL, 16384 | 13880 | LD DE, TEXTA |
| 10260 | LD (AKTPROC), A | 12080 | INC DE | 13900 | CALL #0C0A |
| 10280 | LD SP, (SP2) | 12100 | LDIR | 13920 | SET 5, (IY+2) |
| 10300 | JP PRZPET | 12120 | RET | 13940 | LD IY, (NAVRAT) |
| 10320 | ----- | 12140 | ----- | 13960 | PUSH IY; ADR Z INIT |
| 10340 | AP2 LD BC, 3456 | 12160 | INIT DI | 13980 | LD IY, #5C3A |
| 10360 | LD DE, (N2) | 12180 | LD A, #07 | 14000 | RET |
| 10380 | LD HL, (N1) | 12200 | OUT (#FE), A | 14020 | ----- |
| 10400 | SCF | 12220 | LD A, #3F | 14040 | KONEC DEFB 0 |
| 10420 | CCF | 12240 | LD I, A | | |
| 10440 | SBC HL, BC | 12260 | DEFB 0, 0, 0, 0, 0, 0 | | |
| 10460 | EX DE, HL | 12280 | LD HL, (MEZ) | | |
| 10480 | SBC HL, BC | 12300 | XOR A | | |

DIATEM

Ivo Křepinský, Karlovarská 5, 301 12 Pízeň

V oblasti komunikace člověka s počítačem stále roste úloha dialogu, nejčastěji prostřednictvím displeje a klávesnice. Zvláště mikropočítače jsou pro dialog vhodné již svojí koncepcí. Na druhé straně realizace lepšího dialogového systému není snadná záležitost, navíc je obvykle odtažita od vlastního řešeného problému, proto často programátor vytvoří jakousi náhrádku, která formou dotazů a odpovědí realizuje dialog. Toto řešení je pro programátora výhodné, protože je nenáročné na pracnost; uživateli, a zvláště neznalému uživateli, však přináší obtíže při používání.

DIATEM realizuje dialogovou část programu, umožňuje programátorovi soustředit se plně na vlastní problém, uživateli pak nabízí názorný a srozumitelný dialog.

Výhody DIATEMu

Ze strany programátora:

- méně práce (dialog je hotový),
- srozumitelnější program (dialogová část je oddělena),
- oddělení návrhu displeje a jeho ovládání,
- jednoduchý převod programu na jiný počítač (přízpusobuje se pouze dialogový systém),
- jeden program může zabezpečovat několik činností (při použití přístupových práv).

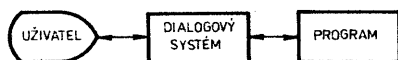
Ze strany uživatele:

- příjemná práce,
- jednotná forma dialogu při různých programech.

DIATEM je realizován pro použití na počítači PMD-85. Jednoduchým způsobem (viz dále) jej však lze přizpůsobit pro libovolný počítač s jazykem BASIC.

POPIS DIATEMu

Dialogový systém zprostředkovává komunikaci mezi uživatelem a programem. Tím je dáno jeho umístění



v systému. To znamená, že veškeré zadávání i veškeré zobrazování se provádí přes dialogový systém. Program je tím jednak zbaven starostí se vstupy a výstupy, jednak se stává hardwarově nezávislým.

Dialog může mít mnoho forem, pro srozumitelnou a jednoduchou obsluhu však jsou nejvhodnější dvě:

- **technika menu** — uživatel si vybírá z nabídky činností (slouží pro řízení činnosti),
- **formulářová technika** — uživatel vyplňuje a opravuje údaje formuláře (slouží pro vstup a výstup dat).

DIATEM umožňuje:

- zobrazení dialogového displeje,
- vstup z položky,
- výstup do položky,
- výběr položky,
- základní editaci zadávaných údajů,
- respektování uživatelských práv pro výběr a zápis.

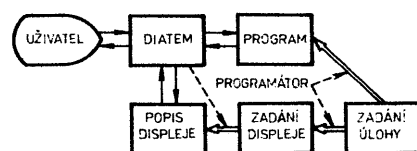
DIATEM je napsán v BASICu v podobě souboru podprogramů, které se připojí k hlavnímu programu.

Základní pojmy

Dialog se uskutečňuje pomocí *dialogového displeje*. Tento displej navrhuje programátor ve tvaru zadání dialogového displeje (pomocí příkazů DATA). Zadání displeje si DIATEM po otevření tohoto displeje při běhu programu mění na vnitřní popis dialogového displeje.

Dialogový displej se skládá z *položek*. S nimi pracuje programátor pomocí funkcí DIATEMu. Funkce umožňují např. vyplnění hodnoty položky, zápis do položky, výběr položky apod. Ke každé položce existují *přístupová práva* pro výběr a zápis. Položka v sobě slučuje vlastnosti bodu menu a údaje formuláře. Má jednotné zadání, popis i funkce, záleží na programátorovi, jak ji využije.

Schéma práce:



- předávání údajů
- vytváření
- realizace vytváření

Přístupová práva

Pro přístup do položek jsou vytvořeny skupiny uživatelů. Každá skupina má jiná přístupová práva.

Uspořádání do skupin je libovolné, každá skupina může obsahovat libovolné uživatele, každý uživatel může patřit do libovolných skupin. Uživatel má pak práva všech skupin, do kterých patří. Rozdělení do skupin určuje programátor.

Při určování přístupu se porovnává množina skupin uživatele s množinou skupin, které mají právo přístupu do položky. Pokud je nalezena alespoň jedna shodná skupina, je přístup povolen.

Informace o skupinách je uložena pozičně, určena polohou bitu ve vektoru; hodnota bitu určuje přítomnost skupiny. Porovnání množin se provádí logickým součinem vektorů. Pokud je výsledek nenulový, nastala alespoň jedna shoda.

Pro speciální typy přístupu lze použít:

- prázdná množina — celý vektor nulový — nikdo nemá přístup,
- univerzální množina — celý vektor jedničkový — všichni mají přístup.

Položky

Položka je z hlediska programu nejmenší, dále nedělitelná část dialogového displeje. Přes ni se pomocí funkcí dialogového systému uskutečňuje dialog.

Položka se skládá z těchto částí:

- **aktivní bod** — bod, na který je možno najet kurzorem, a tak vybrat položku,
- **hodnota** — vlastní hodnota položky (číslo, text atd.).

Každá z těchto částí může chybět, podle toho se rozlišuje použití položky:

| akt. bod | hodnota | použití |
|----------|---------|---------------------------|
| — | + | informace, text chyby ap. |
| + | — | bod menu |
| + | + | údaj formuláře |

Ke každé položce existují dva druhy přístupu:

- možnost výběru aktivního bodu,
- možnost zápisu hodnoty.

Pro každý druh přístupu je v zadání položky uvedena množina skupin, které mají tento druh přístupu povolen.

Pro každou položku může být zadána klíčová klávesa — znaková klávesa, při stisku které se kurzor přemístí na danou položku (urychlení přesunů kurzoru).

Zadání dialogového displeje

Zadání vytváří programátor jako část svého programu pomocí příkazů DATA. V zadání musí uvést:

- neměnné texty na obrazovce,
- zadání položek.

Zadání je ve tvaru:

zadání textu

zadání textu

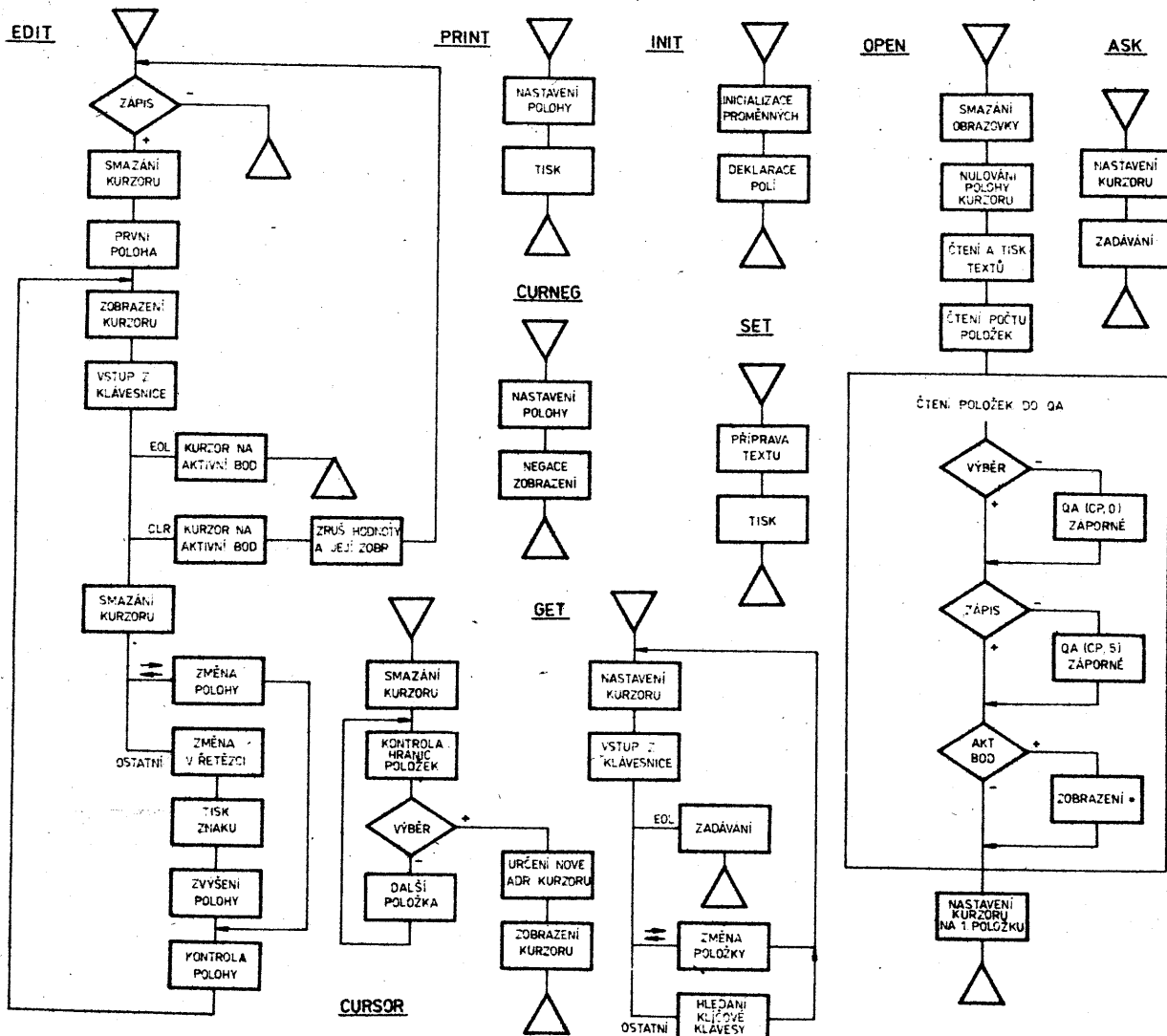
— 1
počet položek (n)
zadání položky 1

...

zadání položky n

Zadání textu je ve tvaru:

vodorovná adresa na obrazovce,
svislá adresa na obrazovce,
text.



Zadání položky je ve tvaru:

vodorovná adresa aktivního bodu na obrazovce, svislá adresa aktivního bodu na obrazovce, klíčová klávesa (znak), přístup k výběru položky, vodorovná adresa hodnoty na obrazovce, svislá adresa hodnoty na obrazovce, délka hodnoty (počet znaků), přístup k zápisu do hodnoty.

Vodorovná adresa klíčového bodu a délka hodnoty jsou v popisu displeje použity k indikaci typu položky, proto musí být nenulová. Neexistence části položky se zadává prázdnou množinou přístupu.

Zadání dialogového displeje může být v programu libovolný počet, jejich rozlišení se provádí před otevřením dialogového displeje příkazem RESTORE číslo řádku.

Popis dialogového displeje

Popis si vytváří dialogový systém při otevření dialogového displeje. Popis je umístěn v poli QA. Je složen z popisu jednotlivých položek, každá položka je umístěna v jednom řádku pole QA takto:

sloupec

- 0: vodorovná souřadnice aktivního bodu (pokud je zakázán výběr položky, je tento údaj záporný),
- 1: svislá souřadnice aktivního bodu,
- 2: klíčová klávesa (ASCII hodnota klávesy),
- 3: vodorovná souřadnice hodnoty,
- 4: svislá souřadnice hodnoty,
- 5: délka hodnoty (pokud je zakázán zápis do hodnoty, je tento údaj záporný).

Popis existuje po celou dobu setrvání daného dialogového displeje na obrazovce (do otevření jiného displeje).

Funkce DIATEMu

Pomocí funkce může program komunikovat s uživatelem. Funkce se volají příkazem GOSUB číslo řádku. Parametry jsou proměnné s vyhrazenými jmény, která jinde v programu nesmějí být použita.

Parametry:

CP číslo položky,
HO\$(CP) hodnota položky,
PP přístupová práva uživatele.

U popisu funkcí je vždy uvedeno jejich jméno (má význam pouze pro lepší porozumění), číslo řádky, použité parametry a činnost.

INIT 51000

Musí být použita před použitím všech ostatních funkcí, a to pouze jednou v programu; slouží pro inicializaci vnitřních proměnných.

OPEN 52000 vstup: PP

Provádí otevření dialogového displeje pro práci; provede zobrazení textů a aktivních bodů a vytvoří popis dialogového displeje podle zadání dialogového displeje a přístupových práv uživatele; před voláním je nutno zvolit dialogový displej příkazem RESTORE.

CURSOR 53000 vstup: CP

výstup: CP

Provádí nastavení kurzoru na aktivní bod dané položky; pokud tato položka nemá povolen výběr aktivního bodu, zkoušejí se postupně následující položky; výstupní hodnota CP označuje položku, kde se kurzor zastavil.

SET 54000 vstup: CP,HO\$(CP)

Způsobí zobrazení řetězce HO\$(CP) v hodnotě položky; řetězec je zarovnán doleva, chybějící znaky doplněny mezerami, přebytečné znaky oříznuty.

ASK 55000 vstup: CP

výstup: HO\$(CP)

Tato funkce umožňuje vstup z dané položky; kurzor skočí na první znak hodnoty a uživatel může hodnotu vyplnit.

Editační klávesy:

←, → posun kurzoru,
CLR smazání hodnoty,
EOL ukončení zadávání.

GET 56000 vstup: CP

výstup: CP,HO\$(CP)

Umožňuje výběr položky a případný vstup z této položky; kurzor je nastaven na aktivním bodu dané položky, uživatel s ním může pohybovat po aktivních bodech klávesami ← (předchozí položka) a → (následující položka) nebo klávesami a klávesou EOL.

vybrat žádanou položku; pokud nemá položka hodnotu nebo je zakázán zápis (bod menu), činnost funkce končí, pokud položka má hodnotu (údaj formuláře), je další činnost stejná jako u funkce ASK.

Program DIATEM

Použití BASIC-G na PMD-85 přineslo několik omezení:

— volání a umístění funkcí:

BASIC-G neumožňuje volání podprogramů jménem, je tedy nutno volat číslem řádku; čísla řádků jsou neměnná, proto je DIATEM umístěn v oblasti 50000—59999, programátor ji nesmí použít.

— parametry funkcí:

BASIC-G neumožňuje předávat při volání parametry, proto byly vybrány proměnné, které program musí před voláním funkce nastavit, funkce je pak použije jako vstupní parametry, podobně funkce nastavuje výstupní parametry.

— lokální proměnné:

BASIC-G neumožňuje lokalitu proměnných, proto mají v DIATEMu všechny lokální proměnné jméno začínající písmenem Q, programátor tato jména nesmí použít v programu.

DIATEM používá tři hardwarově závislé funkce:

— vstup z klávesnice:

je realizován přes podprogram monitoru, volá se funkcí USR(-31583);

— zobrazení kurzoru:

kurzor je znázorněn inverzností zobrazení znaku, používají se pro něj proměnné QB\$, a QJ\$, zobrazování se provádí příkazy BMOVE a BPLOT (podprogram 57200);

— tisk na určenou polohu na obrazovce:

BASIC-G neumožňuje tisk na libovolnou polohu na obrazovce, monitor však alespoň částečně ano; je nutno nastavit adresu na obrazovce do paměti na adresách -16322 a -16321 a nuloval polohu tisku pro BASIC na adrese 24065 (podprogram 57100).

Proměnné

Globální proměnné:

NP maximální počet položek, musí nastavit programátor před voláním INIT,

CP číslo položky,

HO\$(NP) souřadnice kurzoru,

PP přístupová práva uživatele.

Lokální proměnné:

QA(NP,5) popis dialogového displeje,

QB\$ levá polovina kurzoru,

QJ\$ pravá polovina kurzoru,

QC vodorovná souřadnice kurzoru,

QD svislá souřadnice kurzoru,

QE vodorovná souřadnice tisku,

QF svislá souřadnice tisku,

QG\$ řetězec pro tisk,

QH pomocná proměnná, dočasná hodnota,

QI pozice znaku při editaci,

QK směr posunu kurzoru po aktivních bodech.

Pomocné podprogramy

Tyto podprogramy jsou volány z funkcí Diatemu, programátor je však může také použít, pokud chce realizovat speciální chování dialogu. Popis podprogramů je stejný jako u funkcí:

PRINT 57100 vstup: QE,QF,QG\$

Funkce provádí tisk řetězce QG\$ na pozici na obrazovce určenou QE a QF.

CURNEG 57200 vstup: QC,QD

Funkce provádí negaci zobrazení kurzoru na pozici určené QC a QD.

EDIT 57300 vstup: CP,HO\$(CP),QA

výstup: HO\$(CP)

Funkce provádí vstup do hodnoty položky CP s editací.

POUŽITÍ DIATEMU

Pořadí a počet použití dialogových funkcí jsou libovolné, nejdříve je však nutno nastavit největší počet položek NP a provést funkci INIT.

DIATEM neprovádí kontrolu zadání dialogového displeje, ani použití funkcí, to je věc programátora. Pokud se podobná chyba vyskytne, způsobí buď nesprávné chování dialogu nebo ukončení a hlášení chyby BASICu.

Všechny vodorovné a svislé adresy na obrazovce, se kterými se pracuje, jsou takové, jako by byly určeny pro

příkaz BMOVE — vodorovné 0 až 47, svislé 0 až 241 (242 nelze použít).

Přístupové množiny se zadávají jako dekadická čísla reprezentující binární vektor. Vektor může mít maximálně 16 míst, tj. může být 16 různých přístupových skupin. Stejným způsobem se určí přístupová práva uživatele nastavením proměnné PP.

Výběr položky pohybem kurzoru klávesami ←, → se provádí podle pořadí zadání položek. Je proto vhodné zadávat položky v pořadí, které odpovídá logickým pohybům kurzoru.

Přizpůsobení pro jiné počítače

DIATEM je možno přizpůsobit pro libovolný počítač s jazykem BASIC, jedinou podmínkou je možnost realizace

— vstupu z klávesnice,

— zobrazení kurzoru,

— tisku na určenou polohu na obrazovce.

Přizpůsobení spočívá v úpravě těchto tří funkcí:

— vstup z klávesnice — řádky 56060, 57330, s tím souvisejí kódy editačních kláves — řádky 56060, 56070, 56100, 57340, 57350, 57360, 57345.

— zobrazení kurzoru — podprogram na 57200.

— přímý tisk — podprogram na 57100.

```

50000 REM
50010 REM *****
50020 REM *
50025 REM * DIATEM *
50030 REM * DIALOGOVY SYSTEM *
50040 REM * PMD-85 *
50050 REM *
50060 REM *****
50070 REM
50080 REM
51000 REM
51010 REM *** INIT ***
51020 REM INICIALIZACE DIALOGU
51030 REM
51040 PP=-1:QB$="" :QJ$="???????"
51050 DIM QA(NP,5),HO$(NP):QK=1:RETURN
52000 REM
52010 REM *** OPEN ***
52020 REM OTEVRENI DIALOGOVEHO DISPLEJE
52025 REM DISPLEJE SE ROZLIŠUJI POMOCI R
52030 REM
52040 GCLER:QC=0:QD=0
52050 READ QE:IF QE=0 THEN READ QF,QG$:
GOSUB 57100:GOTO 52050
52060 READ NP:FOR CP=1 TO NP:READ QA(CP,
0),QA(CP,1),QH,QG$
52070 IF (QH AND PP)=0 THEN QA(CP,0)=-QA
(CP,0)
52080 QA(CP,2)=ASC(QG$+CHR$(0)):READ QA(
CP,3),QA(CP,4),QA(CP,5),QH
52090 IF (QH AND PP)=0 THEN QA(CP,5)=-QA
(CP,5)
52100 QE=QA(CP,0):IF QE<0 THEN 52120
52110 QF=QA(CP,1):QG$="":GOSUB 57100
52120 NEXT CP:CP=1:GOSUB 57200:GOTO 5300
0
53000 REM
53010 REM *** CURSOR ***
53020 REM NASTAVENI KURZORU NA POLOZKU
53030 REM VSTUP - CP (CISLO POLOZKY)
53040 REM
53050 GOSUB 57200
53060 IF CP<0 THEN CP=NP
53070 IF CP>NP THEN CP=1
53080 IF QA(CP,0)=0 THEN QC=QA(CP,0):QD
=QA(CP,1):QK=1:GOTO 57200
53090 CP=CP+QK:GOTO 53060
54000 REM
54010 REM *** SET ***
54020 REM ZOBRAZENI HODNOTY V POLOZKE
54030 REM VSTUP - CP (CISLO POLOZKY)
54040 REM - HO$(CP) (HODNOTA POLO
KY)
54050 REM
54060 QE=QA(CP,3):QF=QA(CP,4):IF QE<0 TH
EN RETURN
54070 QG$=LEFT$(HO$(CP)+",ABS(QA(CP,5)))

```

```

54080 GOTO 57100
55000 REM
55010 REM *** ASK ***
55020 REM VYZADANI VSTUPU POLOZKY
55030 REM VSTUP - CP (CISLO POLOZKY)
55040 REM VYSTUP - HO$(CP) (HODNOTA POLO
ZKY)
55050 REM
55060 GOSUB 53000:GOTO 57300
56000 REM
56010 REM *** GET ***
56020 REM VSTUP S VYBEREM POLOZKY
56030 REM VYSTUP - CP (CISLO POLOZKY)
56040 REM - HO$(CP) (HODNOTA POLO
ZKY)
56050 REM
56060 GOSUB 53000:QH=USR(-31583):IF QH=1
37 THEN 57300
56070 IF QH=131 THEN CP=CP-1:QK=-1:GOTO
56000
56100 IF QH=133 THEN CP=CP+1:GOTO 56000
56130 FOR QI=1 TO NP:IF QA(QI,2)=QH THEN
CP=QI
56140 NEXT QI:GOTO 56000
57000 REM
57010 REM ** POMOCNE PODPROGRAMY **
57100 REM * PRINT *
57110 REM PRIMY TISK NA OBRAZOVKU
57120 REM QE,QF - POLOHA TISKU, QG$ - TE
XT
57130 POKE -16322,QE+64*QF-256*INT(QF/4)
:POKE -16321,195+INT(QF/4)
57140 POKE 24065,0:PRINT QG$:RETURN
57200 REM * CURNEG *
57210 REM NEGACE ZOBRAZENI KURZORU
57220 IF QC=0 THEN 57250
57230 BMOVE QC-1,QD+1
57240 BPLOT QB$,1
57250 BMOVE QC,QD+1
57260 BPLOT QJ$,1
57270 RETURN
57300 REM * EDIT *
57310 IF QA(CP,5)<0 THEN RETURN
57320 GOSUB 57200:QD=QA(CP,4):QI=1
57330 QC=QA(CP,3)+QI-1:GOSUB 57200:QH=US
R(-31583)
57340 IF QH=137 THEN 53000
57345 IF QH=141 THEN GOSUB 53000:HO$(CP)
="":GOSUB 54000:GOTO 57300
57350 GOSUB 57200:IF QH=131 THEN QI=QI-1
:GOTO 57300
57360 IF QH=133 THEN QI=QI+1:GOTO 57300
57370 HO$(CP)=MID$(LEFT$(QA(CP,5),QI)
+CHR$(QH)+MID$(HO$(CP),QI+1),2)
57380 QE=QA(CP,3)+QI-1:QF=QA(CP,4):QG$=C
HR$(QH):GOSUB 57100:QI=QI+1
57390 IF QI=0 THEN QI=1
57400 IF QI>QA(CP,5) THEN QI=QI-1
57410 GOTO 57330

```